Australian Government
Department of Defence
Defence Science and
Technology Organisation

ORBAT Services Design Version 1.0

Glen Coomber, Carly Forbes and Philip Hawthorne

Command and Control Division
Defence Science and Technology Organisation

DSTO-TR-1727

ABSTRACT

The ORBAT Services Working Group was a joint DSTO, DMO and Industry activity to design, document and prototype a technical information infrastructure for applications and services within the Defence Information Environment that manipulate and store Order Of Battle (ORBAT) information. This document records the results of this activity and provides standardised Web Service Description Language interfaces, XML data representations and Use Cases for the management and manipulation of ORBAT information for use in Command, Control and Intelligence systems.

Published by

DSTO Defence Science and Technology Organisation
PO Box 1500
Edinburgh South Australia 5111 Australia

Telephone: (08) 8259 5555
Fax: (08) 8259 6567

AR-013-413
May 2005

APPROVED FOR PUBLIC RELEASE

ORBAT Services Design Version 1.0

Executive Summary

DSTO was tasked by the Command Support Systems (CSS) Branch of Defence Materiel Organisation (DMO) to apply the ideas and business practices developed in the EXperimental C3I Technical Environment (EXC3ITE) Project JP2061 to applications and services that support Order of Battle functionality. The primary purpose of this activity was to allow the CSS Branch to gain experience with the EXC3ITE business model as a potential method for building an integrated Joint Command Support Environment (JCSE). This document records the results of this effort, in the form of a technical architecture for a distributed system based on Open Component Based Interfaces. This work was carried out in a collaborative effort between DMO, DSTO and Industry staff. The DSTO Task "Re-Useable Order of Battle (ORBAT) Component – JTW 01/322" was used to coordinate these activities.

The primary audience of this document is Systems Architects and Software Developers from the Office of the Chief Information Officer, CSS Branch and Defence Industry. The architecture is of sufficient detail to allow Defence Industry and CSS projects to implement ORBAT systems that are interoperable, whilst still meeting the specialist needs of diverse user communities.

The less technical sections of this document are suitable for end users to gain an overview of the flexibility and technical underpinning of the information infrastructure.

It must be stressed that this document arose from a bottom up technical requirement as part of an effort to explore potential methods of coordination between Command Support Systems projects engaged in the concurrent acquisition of ORBAT tools. Whilst bottom up technical approaches are an excellent way to explore the application of technology to a particular problem space, they do not by themselves adequately consider the business issues that need to be addressed in successfully introducing a distributed system into service. Should the Department wish to formally adopt the ORBAT services, future work will be required to develop a range of supporting architectural products.

Authors

Glen Coomber
Command and Control Division

Glen received a B.Sc (computing) from Griffith University in 1994 and joined DSTO in 1995. He maintains a strong interest in applying distributed systems techniques to Command, Control, Communications and Intelligence applications.

Carly Forbes
Command and Control Division

Carly received a B.Sc (Computing) from Deakin University in 2001 and is currently completing a Masters in IT. Since joining DSTO she has spent majority of her time working with Web Services.

Philip Hawthorne
Command and Control Division

Philip Hawthorne joined DSTO in January 2000 after working for several years as a contractor in the embedded and mobile computing fields. He has a degree in Computer Science and Software Engineering and is currently undertaking a PhD in the field of military information management.

Contents

1. The ORBAT Working Group

The principles of general systems theory assert that adaptable designs are most easily achieved by the use of modular internal components. As self-contained sub-systems, these components need to have open interfaces to allow them to be reconfigured as required. Modern software technology facilitates this at a low level by object-oriented languages. The primary purpose of the Experimental C3I Technical Environment (EXC3ITE), JP2061 was to embody these principles in examples of component-based design extended by middleware services. This approach is termed an Open Component Based Interface (OCBI) methodology, and is believed to provide a cost-effective mechanism for achieving integration of C3I systems. In OCBI the development focus is on the design and definition of reusable software components that have open (well known), non-proprietary, interfaces. To summarise the EXC3ITE position, Defence gains the most value from software and systems when they are:

- Based on common (shared) and reusable software components
- Have well defined and open (non-proprietary) interfaces
- Are designed by a partnership of Defence subject matter experts, DMO project representatives and Defence Industry
- Are supported by a range of architectural products

In order to apply this approach to C3I systems, DSTO recommended the formation of Working Groups in areas where common C3I functionality is identified to undertake and guide the acquisition process. The following diagram shows the role of working groups.

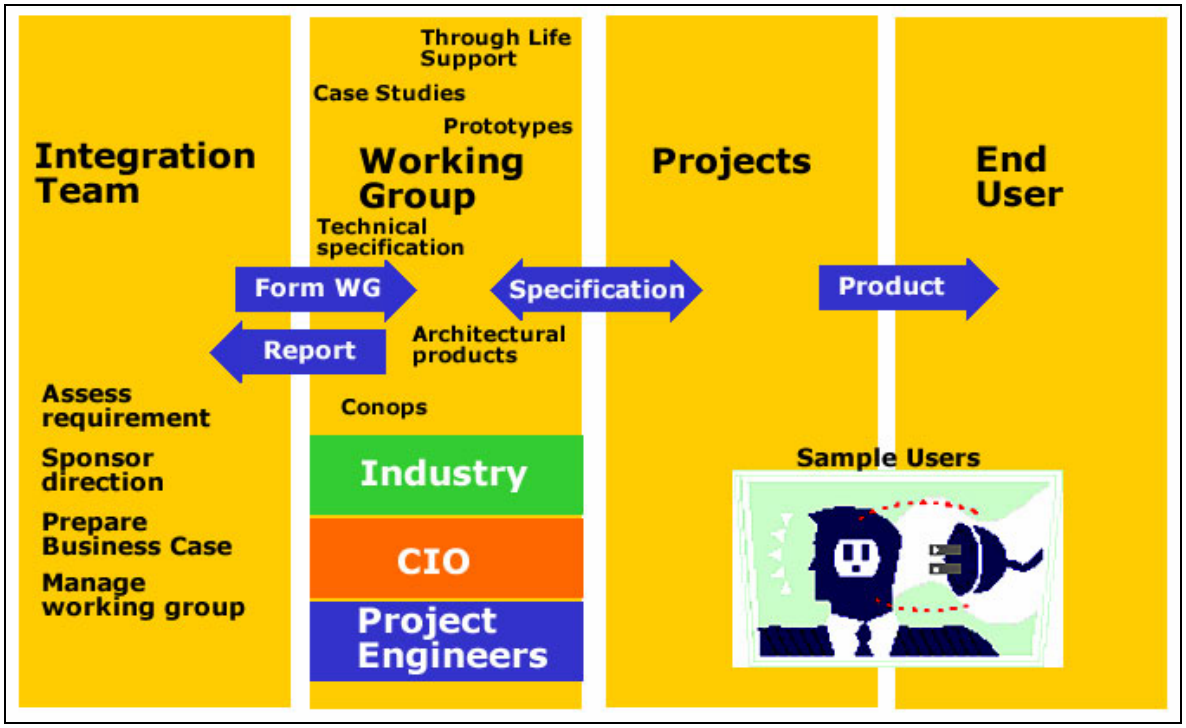


Figure 1 Role of working groups.

In line with this methodology, DSTO has convened and managed the conduct of the ORBAT Services Working Group (OWG). The group was composed of DMO, DSTO and Defence Industry staff working on systems or projects that manipulate ORBAT information. Attendance was open to all interested parties. Throughout the life of a working group the level of interest of individual participants varied significantly depending on the activities that are being conducted at the time.

The participants saw this work as a mechanism to enhance and foster collaboration between parties with a shared interest in the development of specific software components. To date this collaboration has helped the working level staff on the BCSS and JCSS projects to identify common areas of interest and common data standards that should be used in construction of these systems.

The working group met formally on four occasions. During these meetings the concepts, use cases and specifications found in this document have been developed. DSTO staff have undertaken the production of this document and its predecessors to ensure that the results of the working group are well documented. Any errors or omissions are exclusively the fault of DSTO.

Working Groups are not successful if they confine their activities to the theoretical. Past experience in the EXC3ITE project has shown that it is not possible to adequately specify a system to a level of detail that is useful to developers without building a series of prototypes. To avoid this being a purely theoretical exercise, DSTO resourced the development of one prototype with an additional three industry prototypes funded by EXC3ITE. All prototypes were based on a version of the specification.

In addition to allowing us to experiment and further develop the ORBAT specifications, these prototypes established a knowledge base within industry of both the technical and organisational issues associated with ORBAT services. In the case of the JCSS project, arrangements have been made to transition this knowledge and experience into the JCSS development team.

1.1 Role of the ORBAT Services Working Group

The OWG role was to ensure that ORBAT tools and ORBAT data used in multiple ADO systems interoperate to a level acceptable to the CSS Projects.

This report is not meant to describe how to build ORBAT tools or applications to meet specific Defence requirements, only to ensure that these implementations share certain interfaces and behaviours pertaining to:

- Exchange of ORBAT information between ADO Systems
- Exchange of ORBAT information between Allied Systems
- Common information management services including:
    - Advertising availability of ORBAT information
    - Discovery of ORBAT information
    - Identification of ORBAT information
    - Management of distributed information sets
    - Resolution of multiple overlapping information sets
- Linkage of ORBAT information with other ADO corporate information
- Rules regarding access control for ORBAT data

- Rules regarding mandatory metadata, including but not limited to:
    - Currency
    - Volatility
    - Intended Purpose (exercise / MRO / doctrine / etc)
- A Simple Object Access Protocol (SOAP, a.k.a. Web Service) interface to support the most basic ORBAT operations (e.g. how to request part or all of an ORBAT hierarchy).

The OWG believed that the task of defining a definitive data dictionary for all types of ORBAT across the whole ADF is impossible to meet. We therefore concentrated our effort on producing a flexible interface capable of supporting a wide range of underlying data via an extensible data schema. The data schema we have developed may be extended over time without adversely impacting on basic functionality. Where possible we have supported existing data standardisation efforts by adopting data elements from the NATO MIP project.

1.2 Measures of success

The working group will have been a success if the derived interface implementation:

- Is adopted by industry partners implementing ORBAT tools within ADF CSS
- Allows the seamless management and exchange of ORBAT information between CSS
- Enables a basic ORBAT browser built by one vendor to find and display data held in another vendor's service implementation
- Is extensible to meet current and future data requirements
- Does not hinder industry partners from adding additional specialist functionality required by the ADF
- Can be implemented by a competent programmer
- Supports the wrapping of legacy ORBAT databases and systems

1.3 Future Activities

The information infrastructure introduced in this document is very flexible. However, to fully leverage this flexibility this document must be supported by the development of a range of architectural products. These products must be developed in a holistic, system-wide approach. The following are examples of system-wide issues that should be addressed:

- Who will generate ORBAT data for what purposes
- Where data will be stored
    - How it will be maintained
    - System-wide archival policies
    - Required levels of data redundancy
- Integration strategies for the inclusion of Enterprise level information repositories such as PMKEYS and SDSS.
- Integration of the ORBAT infrastructure with other services such as Security.

2. Defining the Problem Space

Traditionally the term Order of Battle has been poorly defined within the ADF. ORBAT is a term that is used loosely in many different contexts to refer to just about any combination of organisational structure with any sort of underlying data. There is a considerable difference of opinion as to how much data and of what type an ORBAT should contain.

The ORBAT Working Group's response to this situation has been to build ORBAT services that support a very wide and easily extensible variety of underlying data. This is combined with much of the simple ORBAT manipulation functionality that will be required by ORBAT applications. We have done this by focusing on providing basic common data representations, which can be extended as needed. These common data representations exist within a distributed (federated and replicated), service based architecture.

In essence the ORBAT Working Group has not sought to solve a particular user's requirement for some level of ORBAT functionality, instead we have focused most of our energy on how to build a flexible infrastructure that can support any user requirement. Considerable effort has been put into understanding and developing an information management infrastructure that will allow us to manage the diversity of data and intended uses for that data within the system. This view of the problem space is quite different to an end user's view; most users are concerned with only one or two applications of ORBATs. On the other hand, we are concerned with all potential ORBAT applications. This difference in scope can cause significant confusion for an end user trying to understand whether these services can help them with their immediate problem. To ease this situation we suggest that end users recognise that this system is an infrastructure much like any other infrastructure such as the telephone network. Lots of effort has gone into designing how the system should work and what happens under certain circumstances. For example, just like with a telephone network, end users can ignore most of this detail and just get on with their specific task.

2.1 Characterising ORBAT Data

There are many potential uses for an ORBAT and many potential applications that can make use of them. Consider a headquarters; they will be interested in an ORBAT view of information for a range of purposes such as:

- Preparing Military Response Options
- Conducting Operational planning
- Conducting exercise planning
- Viewing preparedness information
- Logistical support
- Intelligence
- Force availability

Other users are also interested in ORBAT views:

- As a way to feed simulation systems
- To perform future force capability analysis

With such a large number of uses there is considerable potential to confuse data which is quite similar, but suitable only for a given purpose. This led the OWG to consider how we should characterise and identify the data and its intended purpose. Our solution has been to define a set of common metadata attributes that all ORBATs must have and adhere to. These attributes define the purpose for which the original data was intended (i.e. they are used to decide which data is appropriate for a given purpose).

The metadata attributes are covered in full in the metadata sections (Sect. 4.3 and 6.2.1). However, it is useful at this stage to consider some simple examples to illustrate the problem.

In our first example we consider the case of an exercise enemy force, often referred to as an orange force. We would expect this orange force to have an ORBAT and underlying data associated with it, but we would not want this data to be used for any other purpose except in an exercise. To complicate the matter further, real ADF units will play the role of some components of the ORANGE Force whilst exercise controllers will play the other units. The metadata for this force needs to accurately reflect the intended purpose and the status of each unit and its data. To do this we set up the following metadata. This metadata includes the constraints that would be placed on an ORBAT being used in this context.

Table 1: ORANGE FORCE ORBAT metadata

ORBAT Name:	EXERCISE ORANGE FORCE
Exercise:	True
Derived From Template:	False
Is Template	False
Intended Use:	Plan
Authoritative:	True
Fictional:	True
Effective From:	12/06/2003 (contains the start date of the exercise)
Expires On:	12/07/2003 (contains the end date of the exercise)
Capability Constraint:	OLOC
Unit hostility allowed:	1000006 (Hostile) 1000002 (Assumed Hostile)

3. Introduction to Interfaces, Services and XML

This section is designed as a short primer course to bring users unfamiliar with Information Technology up to speed on the concepts and technologies that underpin the ORBAT Services. A basic understanding of these technologies is useful to fully appreciate the flexibility and adaptability of the ORBAT services.

An overview of the technologies used to complete this project is shown in the diagram below. As you can see, Web Services provide the foundation for the ORBAT Service architecture.

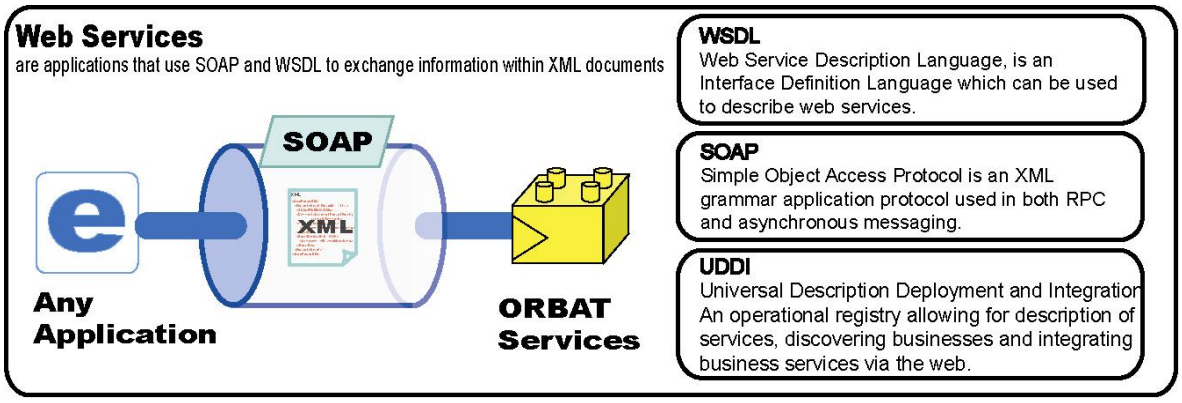


Figure 2 ORBAT Service technology overview

3.1 Interface Definitions

A quick and simple definition for Interface Definition is "*a software description that specifies how to invoke software functions, and what results if any will be returned by those functions*". It describes the software as a black box – what it does in terms of inputs and outputs.

The above definition is well and good but in the real world the heart of the problem has been that there are many software languages, each with their own interface descriptions. There just wasn't an agreed way of getting software to talk. For example, if you wrote an interface description in C, it was specific to the C programming language. You couldn't use a C Interface Description to write Fortran code. The IT industry has been working to address these issues for many years. Several approaches have been tried; for the most part they would have worked if enough people agreed to use them. Unfortunately that wasn't the case. In the next section we describe Web Services and WSDL – this is a common Interface Definition language that uses Extensible Markup Language (XML). It has gained broad acceptance as a standard Interface and is sponsored by the W3C consortium.

Below is a simple example of an Interface – a banking example written in plain text. One interface can describe many software functions and thus do many things. Typically, like functions are grouped together on a single interface. Each software function on an interface is called a method and has parameters for inputs and outputs; many also have error exception handling built in. In the example below we have a method called Get Bank Balance that is used to get the bank balance from a bank account. It takes a bank account number as the input and returns the available funds.

Method: Get_Bank_Balance(bank_account: Target_Account_ID);
Returns: (integer:Cleared_Funds, integer:Uncleared_Funds)
Errors:
Error_Incorrect_Account(string:"An incorrect account code was entered")
End method

The most important concept with Interface Descriptions is that they don't actually do anything by themselves. They only define the inputs and the outputs. For example if two banks (or even better the banking industry) agreed that this interface will be used to query bank balances then each bank would have to implement this interface. Implementing an interface involves writing the backend code that fetches the bank balance from the bank's mainframe or database server and calculates the amount of cleared or uncleared funds. There are automated tools that take an Interface Description and develop the skeleton of your code for you. The advantages of such systems are that much of the drudgery of setting up the system is automated, allowing a programmer to focus their attention on the business logic that implements the interface.

3.2 Web Services

Web Services are a family of protocols that allows many software applications written in different languages to talk together using XML. Most programming tools and platforms now support Web services. These include Microsoft .NET, JAVA and J2EE. In addition, many software vendors (e.g. PeopleSoft) are supplying Web Service Interfaces with their products.

In the previous section we discussed Interface Definitions. In Web Services the interface definition language is the Web Services Description Language (WSDL). This is simply a well-defined way of describing an Interface in an XML file. This WSDL file is parsed by application developers to produce an instance of that service. This means that any system developer can be given a WSDL file and create an interoperable application from that interface description. Of course, developers still need to know about some business logic that applies, which is why we have created this document.

To recap this section so far, a WSDL file defines the Interface to the Web Service, i.e. the functionality offered by that Service in terms of the operations you can perform (methods) and the messages (i.e. parameters) accepted and returned by that method.

3.2.1 Transport Protocols and Message Encoding

A Transport protocol and message-encoding format is used to define the way that clients can talk to a web service. In the ORBAT Services we use the Simple Object Access Protocol (SOAP) transport protocol. This protocol works on HTTP, i.e. it uses the same transport protocol as the WWW. Over time other transport protocols will become available, e.g. in the future, Web Services might use other transport systems such as the MQ series from IBM.

Message encoding refers to how the payload of a method is encoded. We use Document Style encoding in our web services. Document Style encoding means that a well formed XML Document is passed as the Input/Output of a method. One of the benefits of this approach is that many applications already understand XML Documents. Thus you can

pass the output of an ORBAT Service directly to another application without losing the structure of the data. This concept is covered further in the next section on XML Schema.

3.3 XML Schema

As noted in the previous section, Document Style encoding involves the use of XML Documents as the input and output messages for a method. The format for these messages is defined by a technology called XML Schema. XML Schemas are stored in .XSD files.

An XML Schema can be thought of as a document template or a data schema – it sets the design of an XML document. The XML Schema is used to create an XML File (.XML). An XML File is said to be well formed when it contains no errors and conforms to the schemas used in the document.

XML Schema is a lot like making a database that conforms to a particular database design. XML Schema allows the designers of the system to set the overall structure and the allowable properties of a XML document, at a very fine level of detail.

As a simple example, let's say that every document must have a valid security classification, an operation name that is not null and a set of releasability codes. We achieve this by creating a set of XML Schemas that define the different elements. For this example we will need three schema files, one to define the Security Classification, one for the Operation Name and the last to contain the list of country codes that will define the releasability of the document. These will then be imported into another schema that brings them together with the data for an Intelligence Report. We could do all this in just one schema, however if we did that it won't be very easy to reuse the individual elements for other applications. Figure 3 shows the Security Classification Schema in a textual view.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://dsto.defence.gov.au/Security/Classification.xsd"
xmlns="http://dsto.defence.gov.au/Security/Classification.xsd" xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
    <xs:element name="Security_Classification" default="Top Secret">
        <xs:annotation>
            <xs:documentation>Imported from the securityClassification.xsd file</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:enumeration value="Unclassified"/>
                <xs:enumeration value="Restricted"/>
                <xs:enumeration value="Confidential"/>
                <xs:enumeration value="Secret"/>
                <xs:enumeration value="Top Secret"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:element>
</xs:schema>
```

Figure 3 Security Classification.XSD (Text View)

As we can see from the example the values allowed for the Security Classification tag are expressed as an enumerated list containing Unclassified, Restricted, Confidential, Secret and Top Secret. The default value for this tag is set as Top Secret.

The two other schemas we need for our example are MilitaryReleasability.xsd and operation_name.xsd. In the MilitaryReleasability.xsd file the value for the releasability tag is also expressed as an enumerated list, however the tag may be present several times indicating each country allowed to see the document. For example, if a document was classified AUS/CAN/UK/US/NZ a releasability element would appear for each of those countries. The operation name schema is also very simple, it contains a single element which is a text string that holds the name of the operation.

Each XML Schema file has a namespace that uniquely identifies the contents of that Schema; these namespaces can be imported into other schemas to build a larger more complicated schema. To do this we now define an Intelligence Report Schema and import our three base schemas. Figure 4 below shows a graphical example of the Intelligence Report Schema. This is a view produced by XML Spy, the tool that we use to write our XML Schema.

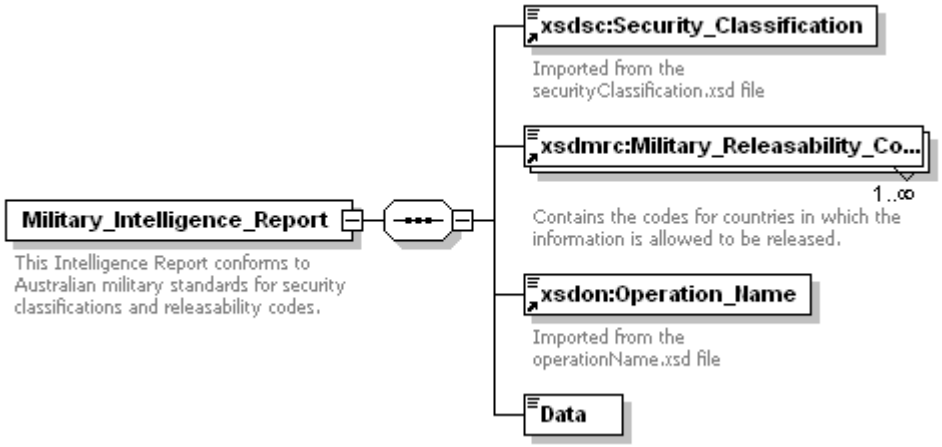


Figure 4 Intelligence Report Schema

The arrow on the bottom left hand corner indicates elements that are references to other schema files. *Xsdon* on the element is a shorthand notation that indicates the element came from the operation_name.xsd file; *xsdsc* indicates that the element came from the security_classification.xsd file and *xsdmrc* indicates that the element came from MilitaryReleasability.xsd. As you can see the Data element is the only one that is defined in the Intelligence report.

For an XML Document based on the Intelligence Report schema to be valid it must have the Security Classification tag present with one value from the enumerated list, at least one releasability tag with a valid value and a string value for the Operation Name, which can't be null.

3.3.1 Interoperability via shared XML Schemas

The good news is that XML Schema is a well-understood and widely implemented technology. All future Microsoft Office applications from the 2003 release onward will talk XML and be able to import and use XML Schemas. Many other programming environments and applications also support this technology. To continue the example, if we were to edit or create an Intelligence Report document based on our schemas above with Microsoft Word, the user will be presented with the security classification tag and

releasability tag shown as drop down fields with the various valid options. The Operation Name field will also be displayed and allow a user to type a name. In this simple example we see how the Microsoft Office 2003 suite understands the Schema. The Office application can be set to only allow the saving of valid XML – thus an intelligence report couldn't be saved unless the mandatory data had been filled out.
There is also a second and vastly more powerful way to achieve interoperability between Schemas using the extensible Style-sheet Language for transformations, which is covered fully in a later section.

3.4 Building a Web Service a simple worked example

Now that we have the basic XML in place we will show how a simple Web Service can be built from these schemas.

Figure 5 shows the process involved in developing an interface based on some .xsd schemas into a web service. As you can see in the diagram, a given service can have several different .xsd files that define the underlying data structures being passed around.

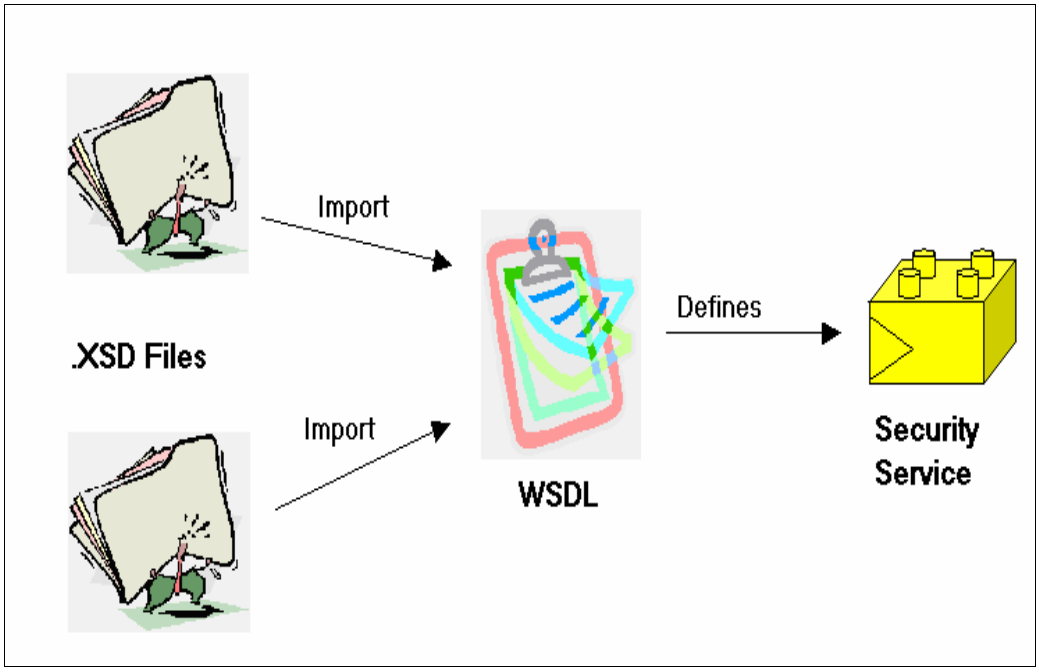


Figure 5 Process for developing schemas into a Web Service interface.

The WSDL file defines what actions are performed and the inputs and outputs required. To illustrate this process we have created a very simple WSDL file that has one operation, checkReleasability. This operation takes the Military Intelligence Report as an input document and returns another Document with a list of country codes that it can be released to. We could extend this operation to perform other functions, for example to check if the data contained the AUSTEO caveat and return an error indicating the document cannot be released.

Figure 6 below, shows the WSDL, the **bold** text indicates where the input for the method has been defined, and the *italic* text indicates the output definitions. First a series of different "messages" are specified. These messages can be used for input, output or both. The bold message definition called "Document" contains an XML document that conforms to the Military Intelligence Report schema we developed earlier. The italics message called "Releasability" defines a series of string elements containing the releasability codes.

```
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:y="http://new.webservice.namespace"
xmlns:ns="http://dsto.defence.gov.au/Security/Classification.xsd"
xmlns:ns1="http://dsto.defence.gov.au/Security/MilitaryReleasability.xsd"
xmlns:ns2="http://dsto.defence.gov.au/Document/security.xsd"
xmlns:ns3="http://dsto.defence.gov.au/Security/OperationName.xsd"
xmlns:ns4="http://dsto.defence.gov.au/Document/MilitarySecurity.xsd"
targetNamespace="http://new.webservice.namespace">
    <import namespace="http://dsto.defence.gov.au/Document/MilitarySecurity.xsd"
location="./MilitarySecurity.xsd"/>
    <message name="messageName"/>
    <message name="Document">
        <part name="Military_Intelligence_Report"
element="ns4:Military_Intelligence_Report"/>
    </message>
    <message name="Releasability">
        <part name="Releasability" element="xs:string" type=""/>
    </message>
    <portType name="SOAPport">
        <operation name="checkReleasability">
            <input message="y:Document"/>
            <output message="y:Releasability"/>
        </operation>
    </portType>
    <binding name="bindingName" type="y:SOAPport">
        <operation name="checkReleasability">
            <soap:operation soapAction="checkReleasability" style="document"/>
            <input>
                <soap:body parts="body" use="literal"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
            </input>
            <output>
                <soap:body parts="body" use="literal"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
            </output>
        </operation>
    </binding>
</definitions>
```

Figure 6 Security Service WSDL

We have now defined a simple security service that uses common XML Schemas to process XML documents and report their releasability based on the releasability schema. At this point we don't have an actual security service, all we have is a description of the inputs and outputs that it takes. To create an instance of the security service we parse this WSDL in the framework of our choice (.NET, J2EE) and generate the framework for the service, the programmer then implements the internal business logic required to perform the methods.

3.5 XML Transformations (XSLT)

In the previous section we saw how XML Schema can be used with WSDL to ensure we have common interfaces working on common data representation, but what happens when we don't have common WSDL or even common Schemas? After all not all organisations can hope to use exactly the same data representations. This is a common problem, there are many times when you will need to change the format of XML Data to some other format, e.g. to a different Schema for use in another service, to plain text, HTML or even a JPEG. The eXtensible Style-sheet Language for Transformations (XSLT) has been created to address these problems.

Like XML, XSLT is a well-defined and broadly implemented standard. XSLT Processes are built into a broad range of products. The role of an XSLT Processor is to take a valid XSLT Document, which defines the operations to be conducted and use it to Transform an input XML Document(s) into some thing else. Figure 7 below shows this process.

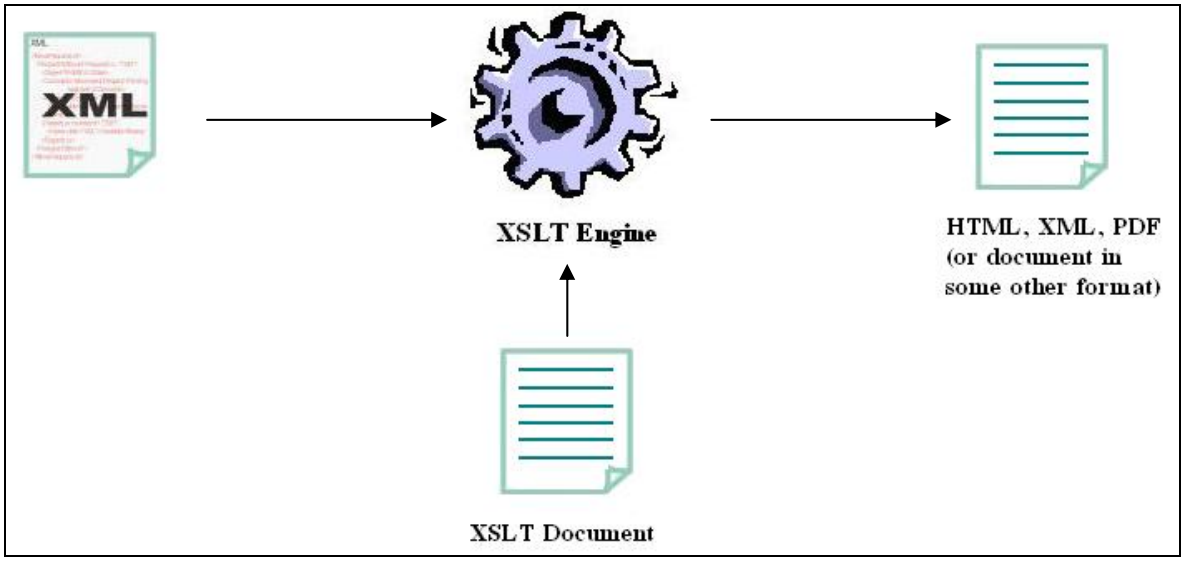


Figure 7 Transformation of XML document into some other format via XSLT

As we can see from the diagram an XSLT Processor is an engine that executes the commands in the XSLT Document to perform some processing on the input document. XSLT is a programming language that is executed by the processor. By defining standard Transforms you can perform standard mappings and complex operations across a range of products.

To illustrate the power and flexibility of this technology we will consider two examples based on the XML Schemas we defined for our Intelligence Report. In the first example we take the Intelligence report and turn it into a web page. In the second example we consider the problem of changing the formatting of a releasable intelligence report to the format required by a coalition partner.

3.5.1 XSLT Example 1: Converting XML into HTML

An important feature of XSLT is that it can be used on a document that conforms to a XML Schema. This means that we can be assured that the input document has all of the fields in

the right order. If any fields are missing or contain invalid values the transformation will not be able to execute. This acts as an additional fail safe to ensure that we don't get corrupted or incomplete output.

3.5.1.1 *The Input Document*

For our input we are using a document based on the security.xsd schemas. It therefore must contain an operation name, security classification and *at least* one military releasability tag.
Figure 8 below contains a valid document with releasability for the document set as AUS/CAN/UK/US/NZ.

```
<?xml version="1.0" encoding="UTF-8"?>
<Military_Intelligence_Report xmlns="http://dsto.defence.gov.au/Document/security.xsd"
xmlns:xsdmrc="http://dsto.defence.gov.au/Security/MilitaryReleasability.xsd"
xmlns:xsdon="http://dsto.defence.gov.au/Security/OperationName.xsd"
xmlns:xsdsc="http://dsto.defence.gov.au/Security/Classification.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://dsto.defence.gov.au/Document/security.xsd
C:\work\Tasks\ORBAT\documentation\ORBATS~1.0\SECURI~1\MilitarySecurity.xsd">
    <xsdsc:Security_Classification>Secret</xsdsc:Security_Classification>
    <xsdmrc:Military_Releasability_Country_Code>AUS</xsdmrc:Military_Releasability_Country_Code>
    <xsdmrc:Military_Releasability_Country_Code>CAN</xsdmrc:Military_Releasability_Country_Code>
    <xsdmrc:Military_Releasability_Country_Code>NZ</xsdmrc:Military_Releasability_Country_Code>
    <xsdmrc:Military_Releasability_Country_Code>UK</xsdmrc:Military_Releasability_Country_Code>
    <xsdmrc:Military_Releasability_Country_Code>US</xsdmrc:Military_Releasability_Country_Code>
    <xsdon:Operation_Name>Op. XSLT Transform</xsdon:Operation_Name>
    <Data>Insert the Intelligence Information here!</Data>
</Military_Intelligence_Report>
```

Figure 8 Security Service XML document example

3.5.1.2 *The XML Transformation Document.*

In order to transform our XML document into a HTML document an XSLT specification is required. The XSLT extracts the Intelligence Information from the XML and places it in the appropriate sections of the HTML document with the necessary formatting tags. An example XSLT is show in *Figure 9* below. This example extracts the security classification and operation name displaying it in large red text across the top of the page. The releasability is shown in green below, followed by the actual data of the Intelligence report in normal text.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xsdintell="http://dsto.defence.gov.au/Document/security.xsd"
xmlns:xsdon="http://dsto.defence.gov.au/Security/OperationName.xsd"
xmlns:xsdsc="http://dsto.defence.gov.au/Security/Classification.xsd"
xmlns:xsdmrc="http://dsto.defence.gov.au/Security/MilitaryReleasability.xsd">
    <!—set the output of the document to be html -->
    <xsl:output method="html" indent="yes"/>
    <xsl:template match="/">
        <!-- Apply Templates to root of the document -->
        <xsl:apply-templates/>
    </xsl:template>
    <!-- Locates the Military_Intelligence_ begins creating the framework for the webpage -->
    <xsl:template match="xsdintell:Military_Intelligence_Report">
        <html> <head>
                <title>Security Service XML to HTML Transform</title>
            </head>
            <body>
                <p><font color="Red">
                        <!-- Applying templates this way allows you to process the document out of order -->
                        <xsl:apply-templates select="xsdsc:Security_Classification"/>
                        <xsl:apply-templates select="xsdon:Operation_Name"/>
                    <hr width="75%" align="center"/>
                </font> </p>
                <center>
                    <b><i><h3>      <font color="#339933">
                        <xsl:for-each select="xsdmrc:Military_Releasability_Country_Code">
                            <xsl:apply-templates select="node()"/>
                            <!-- put / after each country except for the last one -->
                            <xsl:if test="position() != last()">
                                <xsl:text> / </xsl:text>
                            </xsl:if>
                        </xsl:for-each>
                        <xsl:text> Eyes Only </xsl:text>
                    </font></h3></i></b>
                </center>
                <!-- Extract data element and apply formatting -->
                <xsl:apply-templates select="xsdintell:Data"/>
            </body>
        </html>
    </xsl:template>
    <!-- Extracts the operation name -->
    <xsl:template match="xsdon:Operation_Name">
        <center>
            <h1><font color="red">
                    <xsl:value-of select="text()"/>
                </font></h1>
        </center>
    </xsl:template>
    <!-- Get Security Classification -->
    <xsl:template match="xsdsc:Security_Classification">
        <h2><center>
                <!-- Extracts the operation name and places it in [] -->
                [<xsl:value-of select="text()"/>]
        </center></h2>
    </xsl:template>
    <!-- Get the content of the report -->
    <xsl:template match="xsdintell:Data">
        <h3>
            <xsl:value-of select="text()"/>
        </h3>
    </xsl:template>
</xsl:stylesheet>
```

Figure 9 Security Service XML to HTML example XSLT

3.5.1.3 *The Output Document.*

After the transformation has taken place the output document contains the following text

```
<html xmlns:xsdintell="http://dsto.defence.gov.au/Document/security.xsd"
xmlns:xsdmrc="http://dsto.defence.gov.au/Security/MilitaryReleasability.xsd"
xmlns:xsdon="http://dsto.defence.gov.au/Security/OperationName.xsd"
xmlns:xsdsc="http://dsto.defence.gov.au/Security/Classification.xsd">
<head>
    <META http-equiv="Content-Type" content="text/html; charset=UTF-16">
    <title>Security Service XML to HTML Transform</title>
</head>
<body>
    <p>
        <font color="Red">
        <h2>
            <center>
                    [Secret]
            </center>
        </h2>
            <center>
                <h1>
                    <font color="red">Op. XSLT Transform</font>
                </h1>
            </center>
            <hr width="75%" align="center">
        </font>
    </p>
    <center>
        <b>
            <i>
                <h3>
                    <font color="#339933">AUS / CAN / NZ / UK / US Eyes Only </font>
                </h3>
            </i>
        </b>
    </center>
```

Figure 10 XSLT Output document

When the document is opened in a web browser the user is presented with a page that looks similar to the one shown below.

Figure 11 Intelligence Report Shown in browser

3.5.2 XSLT Example 2: Converting between Schemas

XML documents may be required to be passed between several different types of services, requiring the data to be changed slightly in order to be valid under the new service. The diagram below shows the process that would occur for this to happen.

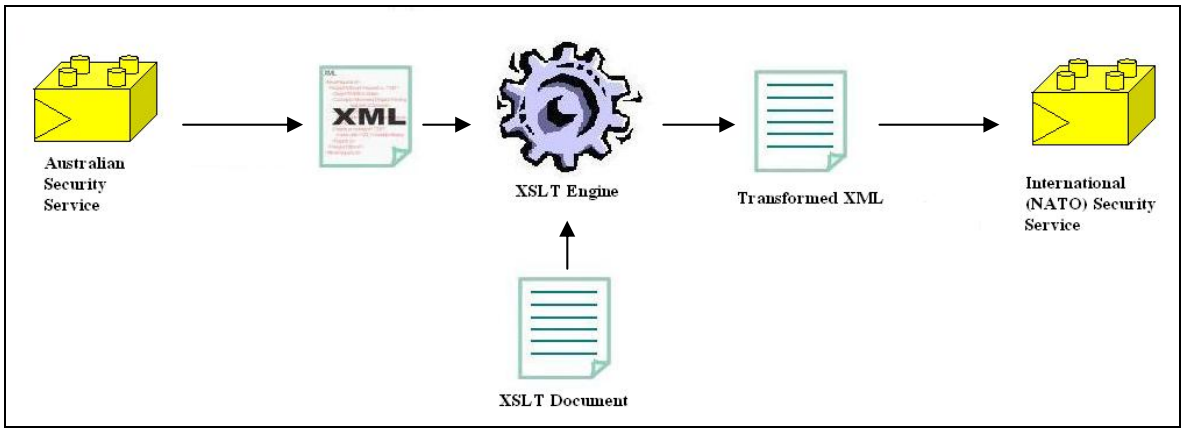


Figure 12 XSLT Process for transforming XML for use in a different service

For example, take the security service we defined earlier, the Releasability codes in the enumerated list for the Military Releasability Country Code tag are defined using a three letter abbreviation. If we wish to pass this information to another International System the country codes may have to be converted to an International standard such as a two letter abbreviation.

Figure 11 shows a sample schema that a System is expecting to receive. We will use this as our target Schema. Note that it has different abbreviations for country codes and the tag name has been changed to NATO releasability country code. Furthermore, the root tag is now called International Intelligence Report not Military Intelligence Report.

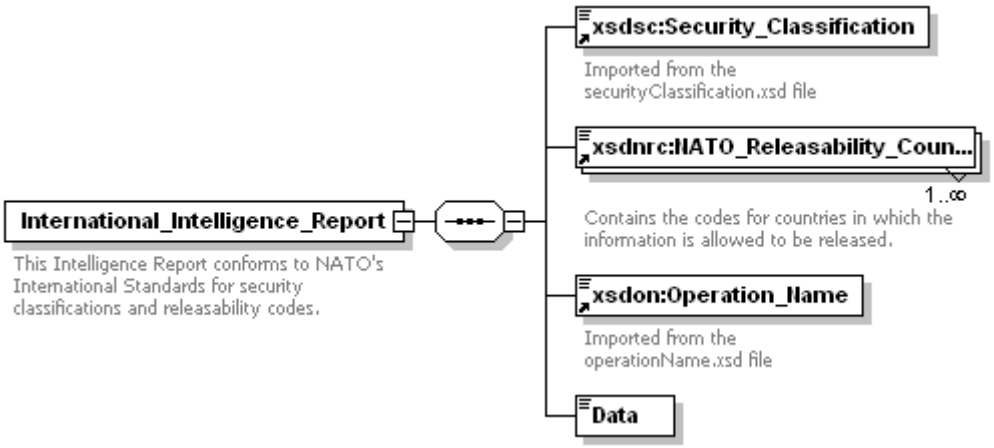


Figure 13 International Security Service schema

You may note that in this example the namespace abbreviation on the elements has changed. This is because the abbreviation is defined in the parent schema when the namespace is imported. Regardless of the abbreviations used both the Intelligence report and the International security report schemas both import the same Security Classification Schema.

3.5.2.1 *Input Document*

The XML that is required to be transformed has been extracted from the Australian security service. It is the same document we used last time.

```
<?xml version="1.0" encoding="UTF-8"?>
<Military_Intelligence_Report xmlns="http://dsto.defence.gov.au/Document/security.xsd"
xmlns:xsdmrc="http://dsto.defence.gov.au/Security/MilitaryReleasability.xsd"
xmlns:xsdon="http://dsto.defence.gov.au/Security/OperationName.xsd"
xmlns:xsdsc="http://dsto.defence.gov.au/Security/Classification.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://dsto.defence.gov.au/Document/security.xsd
C:\work\Tasks\ORBAT\documentation\ORBATS~1.0\SECURI~1\MilitarySecurity.xsd">
    <xsdsc:Security_Classification>Secret</xsdsc:Security_Classification>
    <xsdmrc:Military_Releasability_Country_Code>AUS</xsdmrc:Military_Releasability_Country_Code>
    <xsdmrc:Military_Releasability_Country_Code>CAN</xsdmrc:Military_Releasability_Country_Code>
    <xsdmrc:Military_Releasability_Country_Code>NZ</xsdmrc:Military_Releasability_Country_Code>
    <xsdmrc:Military_Releasability_Country_Code>UK</xsdmrc:Military_Releasability_Country_Code>
    <xsdmrc:Military_Releasability_Country_Code>US</xsdmrc:Military_Releasability_Country_Code>
    <xsdon:Operation_Name>Op. XSLT Transform</xsdon:Operation_Name>
    <Data>Insert the Intelligence Information here!</Data>
</Military_Intelligence_Report>
```

Figure 14 XML extracted from Australian Security Service

3.5.2.2 *The XML Transformation Document.*

The transformation document needs to copy the contents of the Australian Intelligence Report into a new document in the International Intelligence Report format. To do this we to create a new document with the International Intelligence Report tag as its root. The existing data that does not need to be modifies is simply copied to the new Operation Name and Security Classification tags.

The Australian Releasability Country Code tag is different to the International Releasability country code tag. To handle this mapping we have written custom code in the XSLT to map the appropriate code from the three letter Australian format to the two letter International format. For example AUS would become AS after the transformation takes place.

Our example XSLT document is show in *Figure 15*.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xsdintell="http://dsto.defence.gov.au/Document/security.xsd"
xmlns:xsdon="http://dsto.defence.gov.au/Security/OperationName.xsd"
xmlns:xsdsc="http://dsto.defence.gov.au/Security/Classification.xsd"
xmlns:xsdmrc="http://dsto.defence.gov.au/Security/MilitaryReleasability.xsd"
xmlns:xsdnrc="http://dsto.defence.gov.au/Security/InternationalReleasability.xsd">
    <!-- Set the output of the transform to be a XML file. -->
    <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
    <!—Programming stuff to Create Variables to hold the names of new elements and their namespsaces
        Note that these elements use the elements Defined in the Schemas -->
    <xsl:variable name="NATO_Releasability_Element">
        <xsl:text>NATO_Releasability_Country_Code</xsl:text>
    </xsl:variable>
    <xsl:variable name="NATO_Releasability_Namespace">
        <xsl:text>http://dsto.defence.gov.au/Security/InternationalReleasability.xsd</xsl:text>
    </xsl:variable>
        <xsl:variable name="NATO_Releasability_Prefix">
        <xsl:text>xsdnrc</xsl:text>
    </xsl:variable>
    <xsl:template match="/">
    <xsl:element name="International_Intelligence_Report"
namespace="http://dsto.defence.gov.au/Security/InternationalReleasability.xsd">

            <xsl:apply-templates/>
        </xsl:element>
    </xsl:template>
<!—Here is where we copy from the Australian Report into the International report -->
    <xsl:template match="xsdintell:Military_Intelligence_Report">
        <xsl:apply-templates select="xsdsc:Security_Classification"/>
        <xsl:apply-templates select="xsdmrc:Military_Releasability_Country_Code"/>
        <xsl:apply-templates select="xsdon:Operation_Name"/>
        <xsl:apply-templates select="xsdintell:Data"/>
    </xsl:template>
    <xsl:template match="xsdsc:Security_Classification">
        <xsl:element name="{name()}">
            <xsl:value-of select="text()"/>
        </xsl:element>
    </xsl:template>
    <xsl:template match="xsdon:Operation_Name">
        <xsl:element name="{name()}">
            <xsl:value-of select="text()"/>
        </xsl:element>
    </xsl:template>
<!—Here is where we go from three letter codes to two -->
    <xsl:template match="xsdmrc:Military_Releasability_Country_Code">
        <xsl:element name="{$NATO_Releasability_Prefix}:{$NATO_Releasability_Element}" >
            <xsl:choose>
                <xsl:when test="text() = 'AUS'">
                    <xsl:text>AS</xsl:text>
                </xsl:when>
                    <xsl:when test="text() = 'CAN'">
                    <xsl:text>CA</xsl:text>
                </xsl:when>
                <xsl:otherwise>
                    <xsl:value-of select="text()"/>
                </xsl:otherwise>
            </xsl:choose>
        </xsl:element>
    </xsl:template>
    <!—Here is where we copy in the report data -->
    <xsl:template match="xsdintell:Data">
        <xsl:element name="{name()}" namespace="{namespace-uri()}">
            <xsl:value-of select="text()"/>
        </xsl:element>
    </xsl:template>
</xsl:stylesheet>
```

Figure 15 XSLT for transforming XML from Australian security standards to International standards.

3.5.2.3 *The Output document*

After the transformation has taken place the XML will conform to the International schema shown in Figure 13. Notice the country codes are now the 2-letter abreviations and the appropriate elements have been renamed.

```
<?xml version="1.0" encoding="UTF-8"?>
<International_Intelligence_Report xmlns="http://dsto.defence.gov.au/Security/InternationalReleasability.xsd">
    <xsdsc:Security_Classification
xmlns:xsdsc="http://dsto.defence.gov.au/Security/Classification.xsd">Secret</xsdsc:Security_Classification>
    <xsdnrc:NATO_Releasability_Country_Code
xmlns:xsdnrc="http://dsto.defence.gov.au/Security/InternationalReleasability.xsd">AS</xsdnrc:NATO_Releasabilit
y_Country_Code>
    <xsdnrc:NATO_Releasability_Country_Code
xmlns:xsdnrc="http://dsto.defence.gov.au/Security/InternationalReleasability.xsd">CA</xsdnrc:NATO_Releasabili
ty_Country_Code>
    <xsdnrc:NATO_Releasability_Country_Code
xmlns:xsdnrc="http://dsto.defence.gov.au/Security/InternationalReleasability.xsd">NZ</xsdnrc:NATO_Releasabilit
y_Country_Code>
    <xsdnrc:NATO_Releasability_Country_Code
xmlns:xsdnrc="http://dsto.defence.gov.au/Security/InternationalReleasability.xsd">UK</xsdnrc:NATO_Releasabili
ty_Country_Code>
    <xsdnrc:NATO_Releasability_Country_Code
xmlns:xsdnrc="http://dsto.defence.gov.au/Security/InternationalReleasability.xsd">US</xsdnrc:NATO_Releasabili
ty_Country_Code>
    <xsdon:Operation_Name xmlns:xsdon="http://dsto.defence.gov.au/Security/OperationName.xsd">Op. XSLT
Transform</xsdon:Operation_Name>
    <Data xmlns="http://dsto.defence.gov.au/Document/security.xsd">Insert the Intelligence Information
here!</Data>
</International_Intelligence_Report>
```

Figure 16 International Security Report XML

We can now submit this document to an International version of the Security Service. We can be confident that it will work, as it will be validated against the International Schema. We can also be 100% confident that document does not contain any errors in the releasability codes. This is because the input document can be validated against the Australian Schema. If the input document contained an invalid releasability code, such as a country that did not exist it would not be validated and would not be processed further.

This transform can be reused wherever it is needed.

3.6 XML Summary

To summarise this section, the important point to remember is that ORBATs are defined in XML Schema. The ORBAT data structure (what is/is not valid for the fields) is all included in the schema and thus understood by many other applications. If for whatever reason the ORBAT Schema doesn't meet an applications needs then an XSLT Transformation can be used to present the data in the format required. In the next section we expand on the role of Services as the building blocks of a Service Based Architecture.

The implementation detail of the ORBAT schemas is covered in later sections.

3.7 The Role of Services in a Service Based Architecture

A service is a component of software with a well-defined interface that performs an agreed set of functions. Services expose interfaces that hide the details of the processing that is required to perform that function.

Services can have multiple interfaces. To be a Web Service you must use the WSDL interface. There is nothing stopping you having more than one Interface, e.g. Web Services, J2EE and CORBA all present on the same Service.

Services are usually long lived – they are constantly running in the system or automatically started on demand. Many services can run on the same computer. Services can build on top of each other, i.e. one service uses the output of another.

There are several technology frameworks available that let you build services; examples of these include .NET, CORBA and J2EE. All technologies have their own strengths and weaknesses. However the Industry is standardising on Web Services for integration, all the technologies above can also offer Web Service Interfaces. Because we have chosen Web Services we can easily integrate ORBAT data across several technology frameworks.

In Service Based Architectures the services are the basic building blocks. Applications are composed using one or more underlying services. There are many potential services in a C3I system. DSTO has experience in prototyping Imagery Services, Track Services, Geospatial Data Services, Data Diode Services and Theatre Broadcast Services.

3.8 Distributed Services

Distributed services use a *federated approach* to the acquisition and management of data. Distributed services are the *opposite of data warehouses*. In a distributed service architecture there are multiple instances of the service (perhaps written by different contractors in different frameworks) running within the system.

The federated approach allows you to manage data close to where it is created. As a result there is no single centralised repository. Because there isn't a single repository you need a mechanism to perform data federation and information management. Data federation refers to the ability to query a distributed system as if it were a single database. No matter

where the data is held in the system it can be made available. Information management is also critical in these systems - the ability to organize the data and to manage the system as a single entity is very important. If you can't gain an overall view of the system and manage its components remotely, the system will very quickly become unworkable.

In the ORBAT Services we have chosen to federate the system by automatically replicating metadata and selectively replicating data. There are other ways to federate the system; you could choose to run distributed data queries and have the results returned and assembled at the client or you could replicate all data everywhere. We have chosen to automatically replicate the metadata required for locating ORBATs. This allows us to query a local copy of the metadata and discover ORBATs that are held on remote services.

The service that performs metadata replication is the ORBAT Management Service. Clients query this service to find where the data they are interested in is held. The client then goes directly to the ORBAT Service to get the data it needs.

Optional replication of ORBAT Data is also supported within the system. This allows the system to be configured and optimised to suit the individual situation. For example, it makes sense for a deployed system to replicate all the data that might be required for that operation before deployment. Individual updates can then be sent as data changes to conserve bandwidth. In the fixed network, most sites would want to replicate their data to another site for backup and business continuity in case the fixed network connectivity is interrupted. The system's managers may wish to maintain an archive of all data that has been produced during the planning phase of an operation – an Archive server could be established for that purpose.

Figure 17 shows a simple scenario for the replication of metadata and ORBAT data within the system. The administrators have set up metadata replication between the OMSs. The large Lego® blocks represents a OMSs, the smaller Lego® Blocks are ORBAT Services.

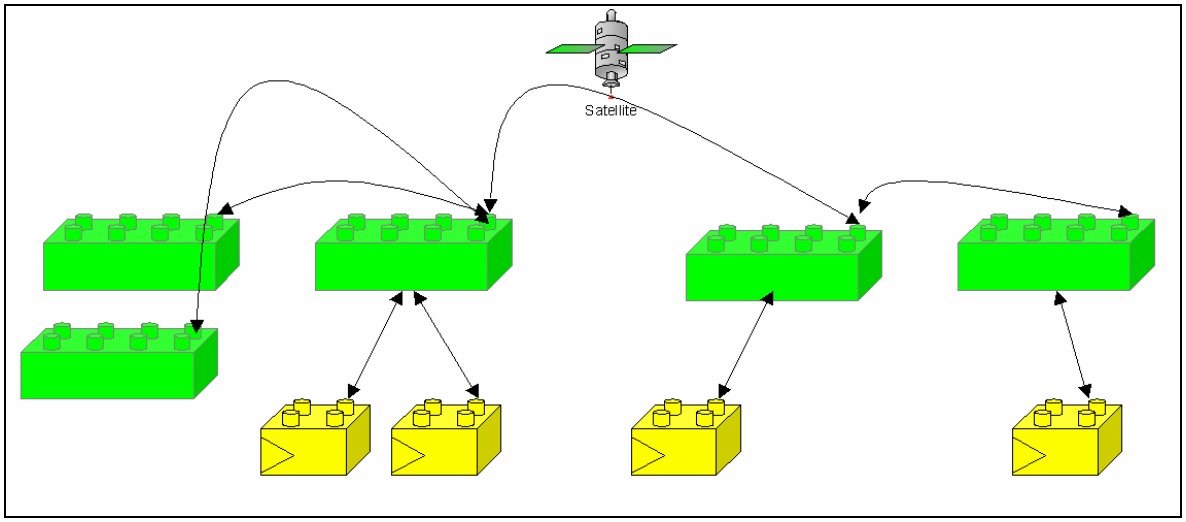


Figure 17 Replication of metadata and ORBAT data.

3.9 Introduction to UDDI

As described in the section above the architecture of the ORBAT Services is distributed. Thus there multiple copies of the same services spread out in various locations. The locations of the services change depending on deployment, etc. Due to the dynamic nature of the locations of the services, we need to be able to locate and identify each service no matter where it is. This requires a locater service to act as a lookup directory. Since we are using Web Services we use the default directory service UDDI.

UDDI stands for the Universal Description, Discovery and Integration service. UDDI is a combination of the white and yellow pages, a variety of different services are registered and stored in a directory, clients and other services can then lookup the directory to find the services they require. For example, when someone wants an ORBAT Service, they simply lookup ORBAT Service in the directory and get a list of all the ORBAT services available, they are then able to choose a service based on the list. This approach is similar to the yellow pages, take mowing the lawns for example, you know you want to hire someone to mow the lawns, but are not aware of what companies are out there, so you look up the yellow pages and are presented with all the services that do lawn mowing. UDDI can also be used to locate a specific service based on the unique ID given to it in the UDDI. Take the lawn mowing example, only this time you know Jim's is the company you want to do you lawns, so you consult the white pages for a specific company to get the details of how to contact Jim's service. UDDI combines both the yellow and white pages in one service. It is a basic "out of the box" commercial middleware service used to find the instance and supported bindings of a Service. UDDI supports finding services by type (e.g. version number of the interface) as well as by organisational owner.

It is important to ensure that a locator service is implemented as shared common infrastructure, which is to say that there would be a single managed UDDI service infrastructure in place (with Defence-wide policies on replication of data, etc). Artificial Defence organisational issues, such as the differences in the responsibility for management of deployed systems versus fixed systems, must not be allowed to interfere with the establishment of this common infrastructure. Failure to observe this would inevitably result in greater cost as each project stands up independent (separately managed) versions of these underlying services.

Consideration is being given to extending the UDDI service to contain optional information about the deployment status of registered services. This would mean that all services that register with the UDDI servers would also have a common deployment status attribute. Moving this information to the UDDI registry would seem to make sense, as many different types of C2 applications need to store information about their deployment status and availability.

4. The ORBAT_DATA Schema

In this section we introduce the ORBAT_Data schema. It is contained in the file ORBAT_Data.XSD. This is the most important schema in the ORBAT service; it defines how all ORBAT data is expressed in XML. A thorough understanding of this Schema is required to appreciate how the ORBAT services represent an ORBAT. All developers should acquaint themselves with this schema before moving on to other schemas used to support individual methods. End users will also benefit from a detailed knowledge of this schema as all ORBAT data is ultimately represented by it.

The ORBAT_Data schema is the message returned as a result of the Get_ORBAT_DATA method. This method returns all data that an ORBAT Service holds for a given ORBAT. Several other methods are also available to return subsets of the ORBAT data. These are used to reduce the amount of information returned to a client. They are designed to return a subset of the information that would have been returned in the ORBAT_Data schema. End users don't need to know which method has been called to get the data; either way the data will still conform to the relevant section of the ORBAT_Data schema.

4.1 Structure of ORBAT_Data

ORBAT_Data is composed of five top-level data elements; Fully_Qailified_ID, METADATA, OrbatDataGroup, ORBAT_CONSTRAINTS, ORBAT_TIMESTAMP.
Figure 18 shows the ORBAT Schema collapsed at the highest level. Each section will be expanded in the following section.

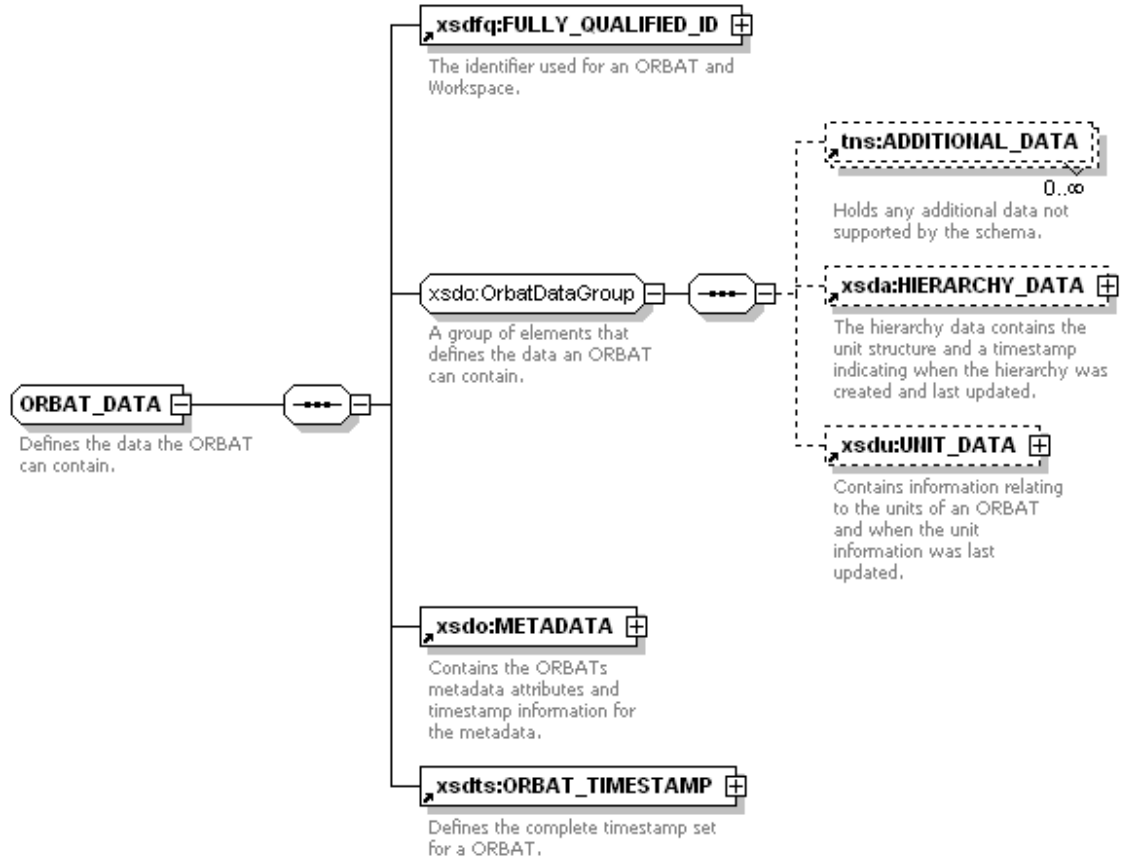


Figure 18 ORBAT data

4.2 Fully_Qualified_ID

This Tag is used to identify the instance of this ORBAT data. In the ORBAT system there can be many copies of the same data each residing on different servers. It is therefore important to know exactly where this data came from. This is achieved by the use of a concatenated key, which is the combination of two sub-tags, the SERVICE_ID and WORKSPACE_ORBAT_ID. Figure 19 shows the elements of a fully qualified ID.

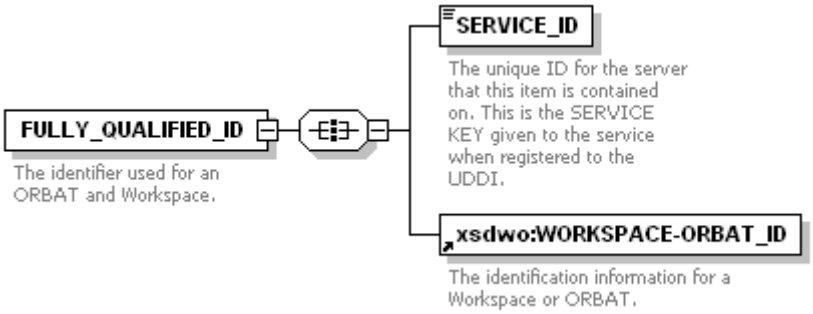


Figure 19 Fully qualified id

The Service ID is a system generated ID that is unique to each ORBAT Service. The ID is generated by the UDDI Service and identifies an instance of an ORBAT Service. This number is not machine specific; an ORBAT service can be located on one computer and later moved to another and the Service ID will stay the same. Software clients use this key to resolve out a binding that identifies the computer that the service is currently installed on. End users can ignore the actual number as GUI clients will resolve this out to a human readable service name for display purposes.

The WORKSPACE_ORBAT_ID is a Globally Unique Identifier (GUID). It is the primary key that identifies an ORBAT or Workspace. GUIDs are an ISO standard and are system generated when an ORBAT is first created. They are guaranteed to be unique to a very high degree of probability.

A partially qualified ID – which is used in other schemas – is simply the WORKSPACE_ORBAT_ID without a SERVICE_ID.

The following two sections describe how these keys are used together to distinguish between Replicates and Copies of an ORBAT.

4.2.1 IDs and Replicated ORBATs

A Replicated ORBAT is intended to be an exact duplicate of another ORBAT. Replicates are used to improve systems reliability and information access times.

A replicate has the same WORKSPACE_ORBAT_ID as the ORBAT it was replicated from; it is an exact replica that is used for the same purpose as the Master ORBAT. A Replicated ORBAT is always placed onto a different ORBAT Service from the Master copy.

The ORBAT Service and the OMS work together to identify one ORBAT as the Master ORBAT. In normal usage, changes should only be made to the Master ORBAT. All ORBATs have timestamp information in their metadata. This is used to track versioning

between replicates. The OMS tracks Master and Replicate ORBATs and can detect if replication conflicts exist. This is covered further in the OMS section.

4.2.2 IDs and Copied ORBATs

A copy of an ORBAT is used to make an entirely new ORBAT from the original. Copies are made because the purpose of the ORBAT has changed and a new activity is being undertaken. A simple and common example of this is that you can make an ORBAT a template; this template is then copied and modified to reflect the scenario that you are working in.

When a copy is made it is given a new WORKSPACE_ORBAT_ID. The history of where the copy was made from is stored in the ORBAT. Unlike Replicates a Copy can be made on the same service.

When Copies are first taken they share all the same data only the ID's are different. This is potentially confusing as now two different ORBATs have the same name and metadata – only the WORKSPACE_ORBAT_IDs are different. To avoid confusion it is a good idea to change the Metadata, especially the ORBAT name and purpose to further distinguish the Copy from the Original. You can then proceed to change the ORBAT Data to reflect the new activity. Most GUI clients will support this in a single operation.

If a GUI has to display two different copies of an ORBAT that have the same name it is up to the GUI to visibly identify them as being different.

4.3 ORBAT Metadata

This Tag contains the metadata about an ORBAT. It has three sub tags that record the metadata values, constraints and timestamp information.

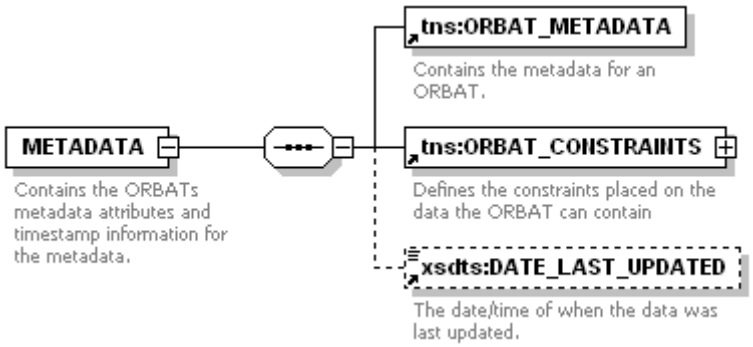


Figure 20 ORBAT metadata elements

The purpose of the metadata tag is to record information required to find and categorise a particular ORBAT. Table 2 shows the attributes of the ORBAT metadata tag.

Table 2: ORBAT metadata attributes

Attribute Name	*Type*	*Use*
Name	String	Required
Exercise:	Boolean	Required
Derived From Template	Boolean	Required
Is Template:	Boolean	Required
Derived From Template ORBAT id	String	Optional
Derived From Template ORBAT name	String	Optional
Intended Use	String	Required
Authoritative	Boolean	Required
Fictional	Boolean	Required
Effective from	Date	Optional
Expires on	Date	Optional

The ORBAT Service passes the metadata, constraints and timestamp information to the OMS where it is replicated and made available for searching. This is addressed further in the Introduction to the OMS.

4.3.1 ORBAT Constraints

This tag is used to define the constraints being placed on an ORBAT. Further constraints can be placed on the ORBAT by adding them to this section of the schema.

The CAPABILITY_CONSTRAINT tag is used to constrain the type of capability data held in an ORBAT (e.g. to ensure that Present Level of Capability (PLOC) and Minimum Level of Capability (MLOC) data is not mixed in the same ORBAT).

The UNIT_HOSTILITY_ALLOWED tag constrains the type of units that may be present in an ORBAT. The hostility codes that are defined under this element are taken from the NATO MIP data dictionary. An example of a constraint would be an ORBAT intended to represent Blue forces. In this case the constraint would be set by choosing the value "1000005"; the documentation for this enumeration is "Friend" – "A unit that belongs to a declared friendly nation."

This element is a repeating element so that a series of constraints can be specified. For example, if the ORBAT was allowed to contain both friendly and hostile units, then the data would contain two instances of the UNIT_HOSTILITY_ALLOWED element, each specifying an individual type. Figure 21 shows the constraints that are currently placed on an ORBAT.

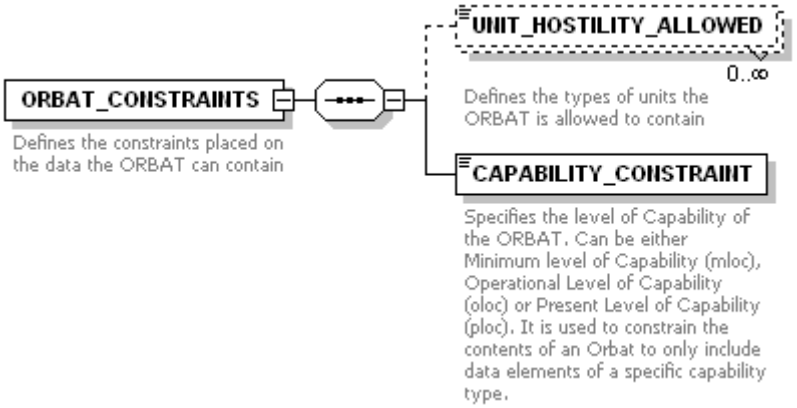


Figure 21 ORBAT constraints

Note that with enumerations, ORBAT GUIs should show the value of the field as "Friend", although in the actual data the value will be "1000005". When a GUI presents a pick list to allow the values to be selected, it would be useful if the definition component were displayed to the user as well as the short name.

ORBAT constraints work in conjunction with Workspace constraints to restrict what data is stored and where within the system. This is addressed further in the Introduction to the OMS.

4.4 ORBAT Data Group

This group contains the elements that make up the ORBAT Data. There are three sections – Hierarchy, Unit Data and Additional Data. Figure 22 shows the ORBAT data group elements.

The following descriptions give a general overview of the contents of this section. Detailed documentation for all data elements has been included in the schemas.

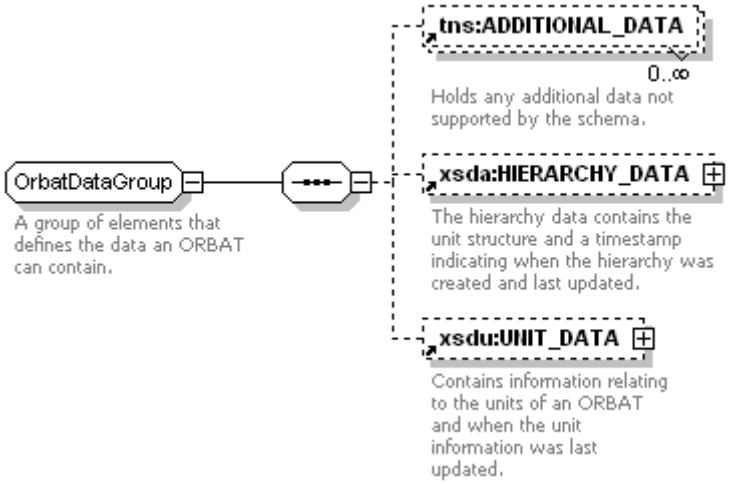


Figure 22 ORBAT data group

4.4.1 Additional Data

The Additional Data tag is used throughout the ORBAT Data Schema to record additional data that is not referenced in the schema. The first instance of this is found in the ORBAT Data Group. This Additional Data tag is used to record a name/value pair of some data type that is required. In the case of this tag the additional data is at the ORBAT level, thus it is intended to record things about the ORBAT.

This structure can be used to record some extra data that needs to be used by some users or systems but not by others. A simple example of this is that an ORBAT logically maps to what is know as a Force in JOLTS. If other applications need to reference the JOLTS ID for a particular force it would be exposed here as a name/value pair. The name would be something like "JOLTS FORCE IDENTIFIER" with a value of "1000294".

4.4.2 Hierarchy

The Hierarchy of an ORBAT represents relationships between units within an ORBAT. There can be zero or many hierarchies per ORBAT. An ORBAT without any Hierarchies present would be displayed as a flat structure. An ORBAT with multiple Hierarchies will appear as if it had multiple root units.

Each Hierarchy has a Root unit; this is where the hierarchy starts. All units under this Unit are stored as attachments. Each Attachment has a state of command attribute that describes the relationship to its parent unit. Values for state of command are derived from the NATO MIP data dictionary. Each attachment also has a start date and an end date to indicate when the command relationship is valid.

The NATO MIP data model also includes a comprehensive event model. At this stage this has not been integrated into the Hierarchy. This will be addressed at some later point. When introduced it will allow units to be added or removed on an event as well as a date time.

Unit information stored in the Hierarchy is extremely limited. The only data stored here about a unit is the EID – the global unique number that identifies a unit. If a unit in the Hierarchy is also included later in the unit data – or available from an ORBAT within the Workspace view, the name for the unit will be displayed using that data. In some cases a client may have to resolve the unit name and display details from the Unit ID Service using the EID. Unit Identity Issues are covered in the Introduction to the Unit ID Service.

4.4.3 Unit Data

The Unit Data contains a repeating set of Unit Tags that store information about a unit and its identifiers, billets (positions and people) and equipment data. Figure 23 shows the unit data.

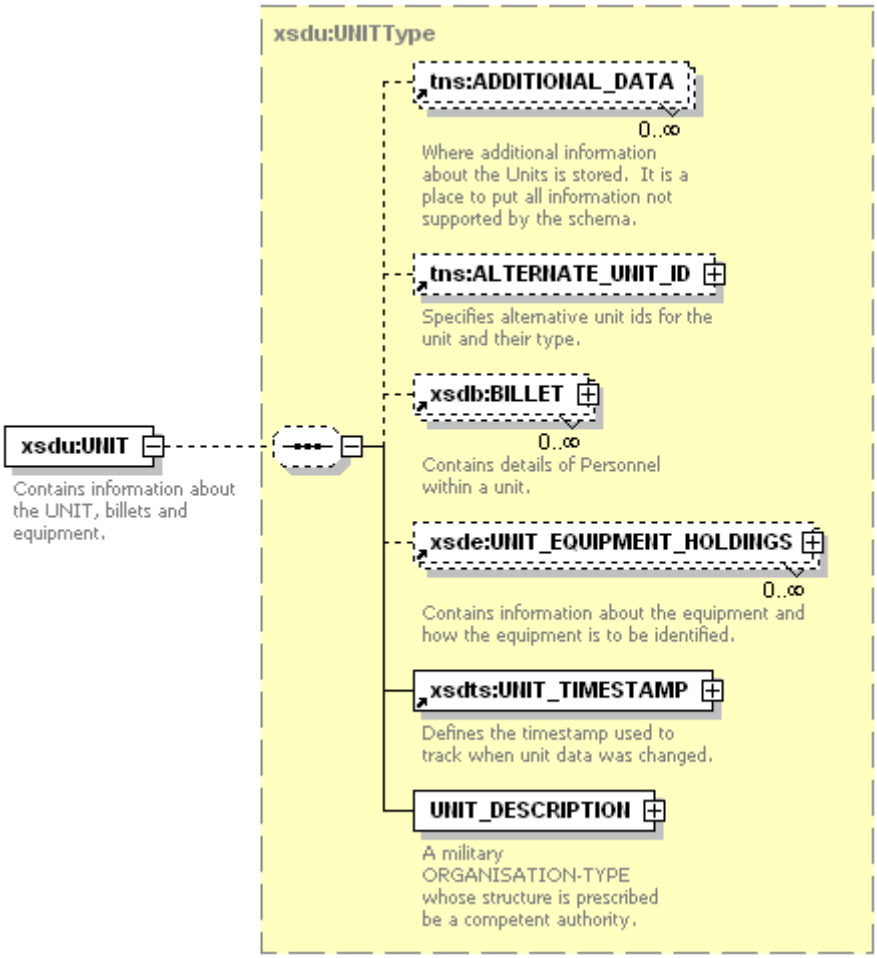


Figure 23 Unit data elements

4.4.3.1 Unit Tag

Each Unit Tag stores data for one unit. A unit is defined as any element of a unit hierarchy that has structure. A Unit in the ORBAT Services is not at the same level that we might think of as a unit in the ADF. ORBAT units are much smaller, typically down to the squad level. The attributes of the unit tag are shown in Table 3, below.

Table 3: Unit Attributes

Attribute Name	*Type*	*Use*
Eid	String	Required
Name	String	Required
Abbrev	String	Required

Each Unit may have the following information included:

4.4.3.2 Alternate Unit ID

The Alternate ID tag holds other unit IDs from other systems that can be used to identify the unit. This tag allows interoperability to be built between systems by building up a list of all known alternate IDs for a unit. This information will be hidden by most GUIs, as they will display only the IDs relevant to their system.

4.4.3.3 Billets

In this tag information is stored about the positions and potentially the people that are attached to this unit.

4.4.3.4 Unit Equipment Holdings

In unit equipment holdings we store the number and type of equipment assigned to a unit. Note that a capability type is used to indicate the sort of equipment data that is stored (e.g. if the ORBAT represents the critical people and equipment for a particular Military Response Option (MRO) the data would be MLOC).

4.4.3.5 Unit Description

The unit description tag is used to store a range of information that has been taken from the NATO MIP. The information stored ranges from the hostility of the unit to data like the size (i.e. brigade or company) and type of unit (i.e. headquarters, support unit or combat unit).

5. Introduction to the ORBAT Service

The ORBAT Service is the main service in the system. It is responsible for importing and exporting information held in an underlying database or system. The ORBAT Service transforms the underlying ORBAT data representation, e.g. a relational database, into the ORBAT Data Schema representation. In this way legacy systems can be integrated with the ORBAT Service.

The ORBAT Service is used in two distinct roles; these are the maintenance of ORBAT data and the replication of ORBATs between ORBAT Services. Figure 24 shows a client maintaining ORBAT data. The client can get data out and put data in via XML. The ® blocks represent ORBAT Services.

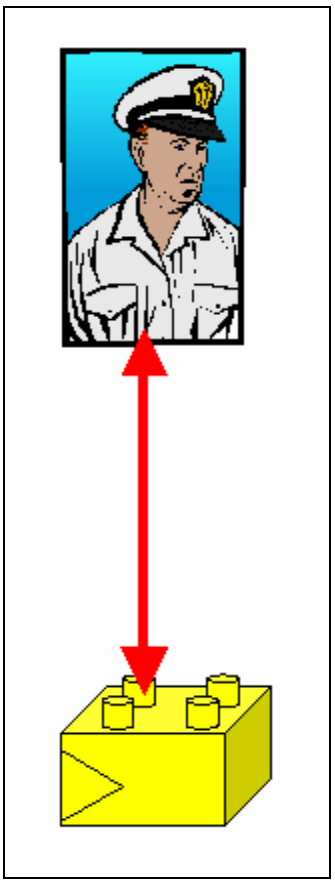


Figure 24 Client maintaining ORBAT data

Figure 25 shows a client managing the replication of data between ORBAT Services. The client tells an ORBAT service to replicate/copy an ORBAT to another ORBAT service.

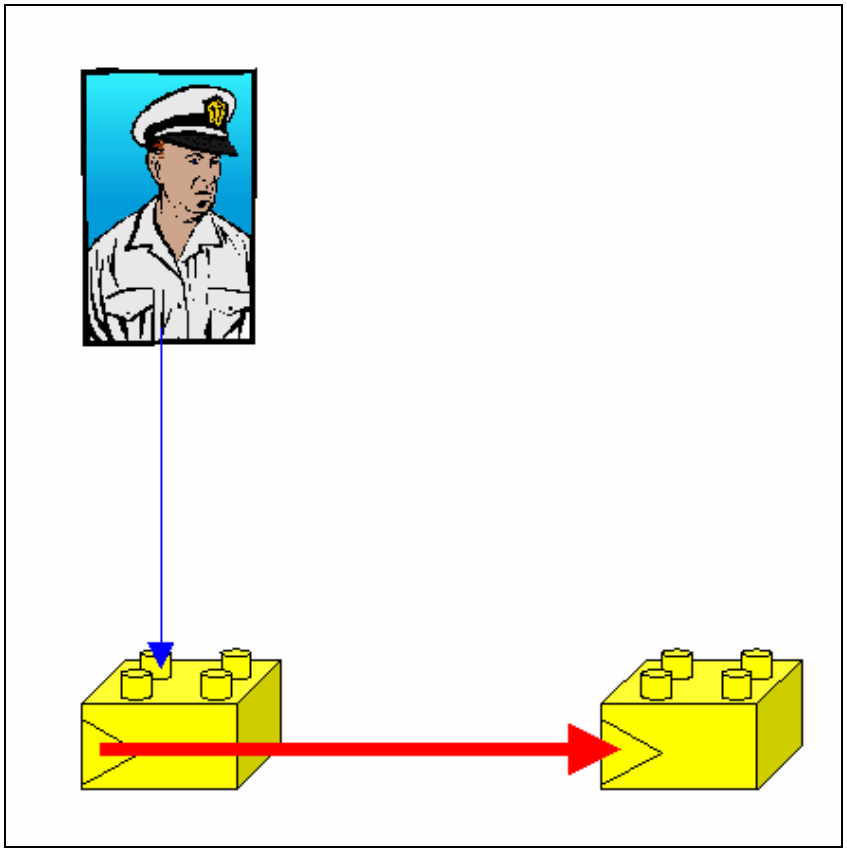


Figure 25 Client managing the replication of data between ORBAT services.

In each of these roles the ORBAT Service is supported by and interacts with the ORBAT Management Service. Details of ORBAT Service to ORBAT Management Service interactions are covered in the Introduction to the OMS.

5.1.1 ORBAT Clients

An ORBAT client interacts with both the OMS and the ORBAT Service. Clients use the OMS to locate a copy of the desired ORBAT and then request the data from the appropriate ORBAT Service. The client may then view and manipulate the ORBAT data. The client is responsible for modifying the contents of the data and returning that data back to the ORBAT Service. This is similar to taking a document from a document repository, editing it in a word processor and returning it to the document repository. The document repository knows very little about the internal structure of the document. In the case of the ORBAT Service, several coarse business logic checks are performed to ensure that the ORBAT is valid.

5.2 ORBAT Service Activities

Clients use the ORBAT Service to:

- Create a new ORBAT
- Update an ORBAT's Data

    - The entire ORBAT can be updated in one hit or individual sections of ORBAT can be updated independently.
- Return ORBAT data
    - The entire ORBAT can be retrieved or individual sections may be retrieved independently
- Copy an ORBAT
    - Copies can be made on the same service or to another service
- Replicate an ORBAT to another service

The ORBAT Service does not support deleting ORBATs directly; all deletes are passed through the OMS to ensure that system-wide data retention policies are enforced.

6. Introduction to the ORBAT Management Service (OMS)

This service acts as a distributed index for ORBAT metadata. It also stores and manages Workspace metadata. The main role of the OMS is to federate with other OMS to provide accurate and timely metadata about the data held in multiple ORBAT Services.

ORBAT Services register with only one OMS. An OMS may have several ORBAT Services managed by it. When an ORBAT service registers with the OMS, it loads and then continuously updates the metadata and timestamp information for all ORBATS contained on that ORBAT service.

Clients search for ORBATS and Workspaces via an OMS. The OMS contains all the metadata for ORBATs and Workspaces; this provides enough information for a client to search on. The client is responsible for determining the most appropriate data for its purpose. When a client has located the most appropriate data it goes directly to the appropriate ORBAT Service and retrieves the data.

OMSs have federation relationships between them; these set the replication and forwarding rules for metadata to flow between OMSs. Typically these relationships are set via the system's administrators. Federation controls how much data is visible to different parts of the overall system. Federation is used in conjunction with replication of ORBAT data to tailor the system to meet particular requirements.

As a future enhancement, the OMS could be extended to automatically include full network quality of service parameters. This would then provide enough information to ensure that a client could rank data according to its metadata and the cost of network access to the resource. For example, replicas of ORBATs that are accessible via high cost links such as satellite can be ranked lower than those available on low cost links such as the local LAN. If there is sufficient interest and a strong business case to do this, it would be sensible to consider the required network metrics as a separate service (as many other applications may require this information); perhaps the UDDI registry or corporate x500 directory could be used to record this information.

6.1 OMS Activities

The OMS performs the following activities in conjunction with the ORBAT Service:

- Tracking the availability of managed ORBAT Services
    - Tracking the status of managed ORBAT Services - is online, is scheduled offline, unexpectedly offline, deployed.
    - Scheduling information includes tracking and identifying planned periods of shutdown due to maintenance or because the service is currently being deployed.
- Providing and managing Workspaces
- Managing the locations and metadata of ORBATs
- Providing a query interface to locate ORBATs and Workspaces and retrieve their metadata

- Replication of ORBAT and Workspace metadata to increase overall systems availability
- Management of replicated ORBAT data sets

The OMS also perform several activities to create and maintain an OMS federation:

- Aggregate and forward metadata update messages
  - Cache and resend updates to overcome network unavailability.
- Track replicated copies of ORBATs
- Identify replication conflicts

6.2 Introduction to Workspaces

A major part of the OMS functionality is the management of Workspaces. These are used to capture the relationships between ORBATs and the context in which they are used. Workspaces are used to assist in grouping and managing collections of ORBATs.

The simple definition of a Workspace is a logical container that holds and or references a number of ORBATs and Workspaces. Workspaces group together ORBATs and other Workspaces that are used for the same purpose. Workspaces have rules and structures to create relationships between them. They perform a similar purpose for ORBATs as directories do for files in a file system.

Figure 26 shows a Workspace that has been created to manage the planning for an ADF Exercise. The main exercise Workspace contains two child Workspaces called RED FORCE and BLUE FORCE. These two Workspaces are responsible for grouping together the ORBATs that will be used as enemies and friendlies during the exercise. For the purpose of this example, let's say the exercise consist of two different scenarios; both the RED FORCE and BLUE FORCE Workspace would contain two ORBATs specifying the force structure for each scenario. The Workspaces are indicated in **bold** text with a dark background, the ORBATs are indicated in plain text with a white background.

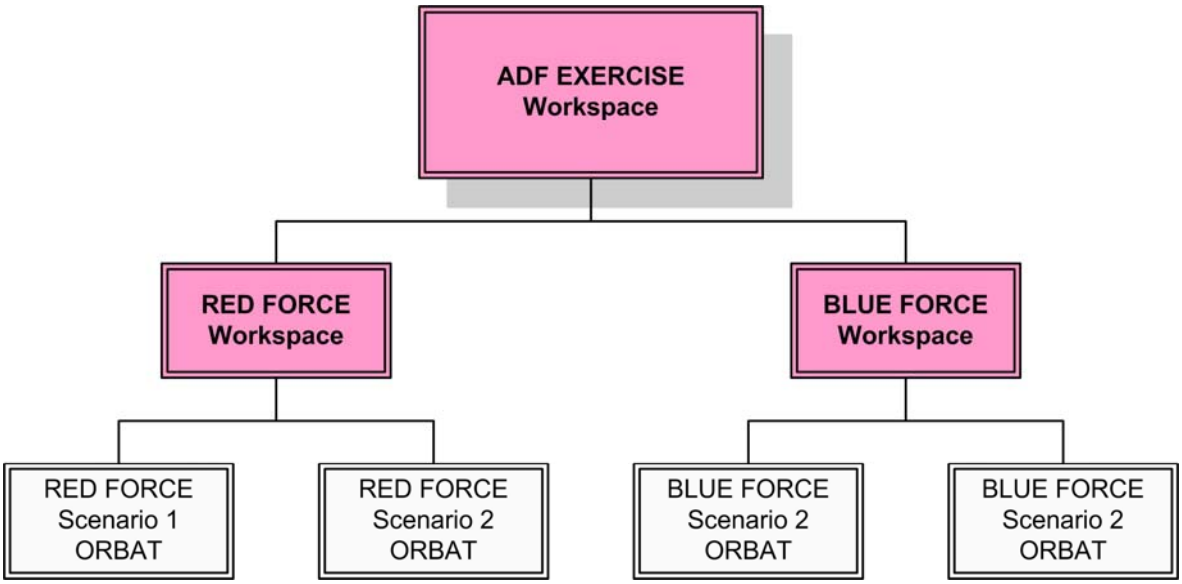


Figure 26 ADF Exercise Workspace

6.2.1 Workspace Metadata

Workspaces have similar metadata to ORBATs and Units. Workspaces also have constraints that apply to sub Workspaces and ORBATs. Figure 27 below shows the three elements that make up the metadata for the Workspace.

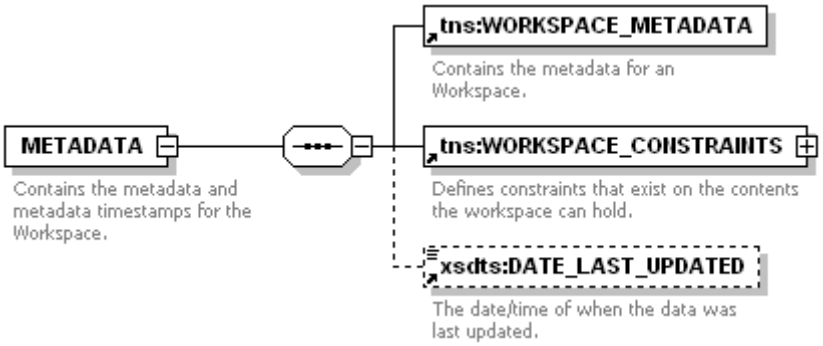


Figure 27 Workspace metadata

Unlike ORBATS, all Workspace metadata is held on the OMS.

6.2.2 The relationship between ORBATs and Workspaces

There are two types of relationships between ORBATs and Workspaces. These are the *managed by* relationship (a.k.a. parent/child relationship) and the *linked* relationship. To explain further:

An ORBAT is *managed by* one and only one Workspace. The Workspace that manages an ORBAT is referred to as the ORBATs *parent* Workspace. The Parent Workspace is responsible for:

- Recording the locations of all Replicas of child ORBATs
- Determining which version of an ORBAT is the Master Copy
- Recording security permissions etc. (to be implemented later)
- Managing Views of data in the Workspace (Views are covered in the next section)
- Holding a duplicate of the ORBAT metadata and timestamps for searching purposes

ORBATs and Workspaces may also be *linked* into another Workspace. A link is a symbolic reference or pointer to another ORBAT or Workspace. You can think of it as a shortcut to where the data resides. The important point to note is that ORBAT data is linked to a Workspace but is not managed by that Workspace.

6.2.3 Workspace View Management

Workspaces contain an optional view specification to define which data should be used to construct a user-level ORBAT. Clients load this view and display the data according to its rules. Views are a new addition to the ORBAT Services and have not been fully prototyped as of Version 1.0.

7. Introduction to the Enterprise Identifier Service

The interoperability of stateful entities hinges on (at the very least) global persistent identification of the said entities. To this end, the US Army Research Labs have conducted research that concludes a 64-bit enterprise identifier (EID) is the best approach for this task. As part of the ORBAT Services Task, DSTO has implemented a concept demonstrator of this service, which is used to create and maintain Unit Identifiers.

The Unit EID service is responsible for the generation and registration of Unit IDs. It also records some simple attributes of a unit such as its name and alternate names and a type, which identifies it as a real unit, fictional unit for training or a template unit.

This service will ensure that the IDs are unique. It will allow a reverse lookup, i.e. given a name it will return a unit ID. The service may support the recording of name changes through time.

This service includes at least one service per environmental command. This will allow delegation for concepts such as the ability to delegate a range of identifiers to a specific server. This operates in much the same way as the Domain Name Service. It is expected that instead of having one EID service there would the creation of Identities to the appropriate level.

The service itself is very simple – each instance of a service should have a region of responsibility, and a unique EID seed (the first 32 bits of the EID). The importance of a region of responsibility becomes apparent when multiple instances of the service are running in a disconnected environment, and two requests are made for an ID for the same thing, such as a unit.

Each service will have a database of entries (with a common EID seed) that can be replicated with other services in slow time. The database design will match the interface (substituting appropriate types), although some provision may be necessary for time stamping entries.

7.1.1 Service Definition

The EID Service is contained in the file EID.WSDL. Figure 28 shows the WSDL file for the EID in a graphical view.

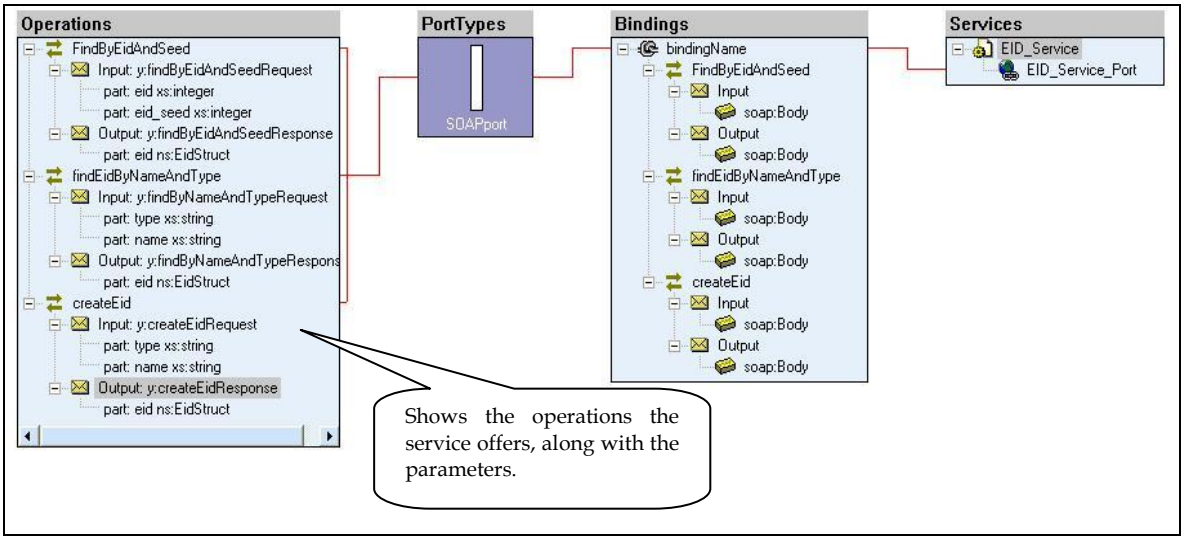


Figure 28 EID service wsdl - graphical view

The WSDL specifies three methods that are available on the EID Service. There are two search operations and a create operation. Each operation returns an eid structure, which contains the eid, eid_seed, type and name. Figure 29 shows the elements returned when accessing a method on this service.

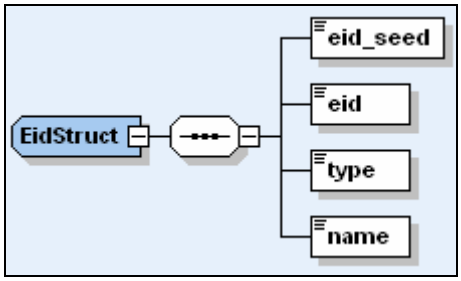


Figure 29 EID Structure

7.1.2 Expected usage

Users creating new units for use within a hierarchy would use this service. It is anticipated that for ADF units this will be a fairly intensive once-off process controlled by the environmental HQs.

Thereafter most users would query the service to select units. An alternative approach would be to have a Doctrinal ADF ORBAT Workspace. This Workspace would contain the standing ADF ORBAT. Users could select Units from this Workspace instead of constantly referring to the Organisational ID service. Either approach is valid, with the latter being a viable option to create a standard view of the ADF structure.

After a unit is created it would be expected that both the ID and the unit name would be stored in the ORBAT service for ease of reference. However, the Organisational ID service would always be considered the Authoritative source for Unit Identity.

System managers may use this service to load lists of available units into their particular instance of an ORBAT tool. This approach would allow light weight ORBAT clients to use Organisational IDs without needing to connect to a running service.

Some consideration needs to be given as to how to use this service for Red forces. This will be important, especially for tactical level intelligence. At this level a large number of new units can potentially be created simultaneously. Consideration may be given to establishing a Red Force Workspace for an operation or alternatively a series of Red Force ORBATs within the Operations Workspace. Ultimately such questions must be answered by Doctrinal experimentation in order to establish the best method of managing this information.

8. Use Case and Sequence Diagrams

The following use cases have been developed to aid discussion of ORBAT issues. They should not be taken as a definitive statement of the user requirements for a particular tool or system. They seek only to explain how a series of ORBAT tools might be used within the system. Readers should attempt to focus on how the systems may be used, rather than the fine detail of the use cases. This is particularly important, as there are many ways to deploy and use these services. How the final system is deployed and configured will be dependant on the business needs of Defence, which of course change over time.

Each use case contains diagrams representing interaction with the system and the sequence of events that must take place to complete the specified operations. These diagrams only show the process flow when operations have been successful; errors and exceptions have not been shown.

The metadata for the Workspaces and ORBATs encountered in the use cases has also been defined.

N.B. Only metadata that is pertinent to the scenario is shown. There are other optional attributes that may be specified depending on the context of the Workspace or ORBAT. The complete metadata set for ORBATs and Workspace is defined in Sections 4.3 and 6.2.1.

In use cases where Services do not need to be sited at a specific location (as with the MRO related ORBAT Service Usage use case) the diagrams use a generic ORBAT Service and ORBAT Management Service. When an ORBAT is copied from one service to another, one ORBAT Service will be called “Source ORBAT Service”, indicating where the ORBAT is being copied from and the other service will be called “Destination ORBAT Service”.

8.1 Land Headquarters Force Assembly

This use case describes a tool that is intended to assist a military planner in assembling a number of forces for a series of theatre operations.

Preconditions:

Existence of a *Doctrinal* Workspace called ADF DOCTRINAL. This Workspace contains the structure of the totality of all forces potentially available. It is decomposed into several child Workspaces, (ARMY Doctrinal, NAVY Doctrinal, AIR Doctrinal). Within each of these Workspaces there will be a number of ORBATs. Relationships between the ORBATs and their units would be hierarchical, so that forces would be composed into larger forces in the standard way.

Access Control is maintained on this Workspace. The LHQ planner has read-only access to the ADF DOCTRINAL Workspace.

The Land Headquarters (LHQ) ORBAT Service has established a relationship with the JCSS Fixed ORBAT Management Service. Figure 30 below shows the deployment of the Land Headquarters ORBAT Service.

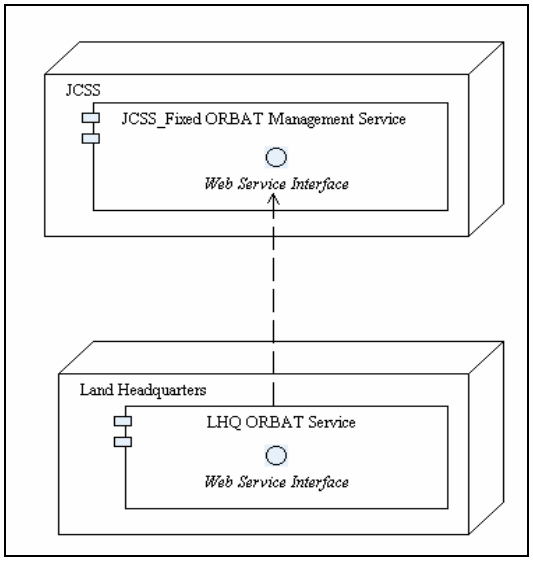


Figure 30 Land Headquarters Deployment

Operations to occur:

1. Create planning Workspace for Force Assembly.
2. Find ADF DOCTRINAL Workspace.
3. Populate planning Workspace for Force Assembly.
4. Create operational Workspace.

Figure 31 is a simple representation of a series of Workspaces that are used and created for this scenario.

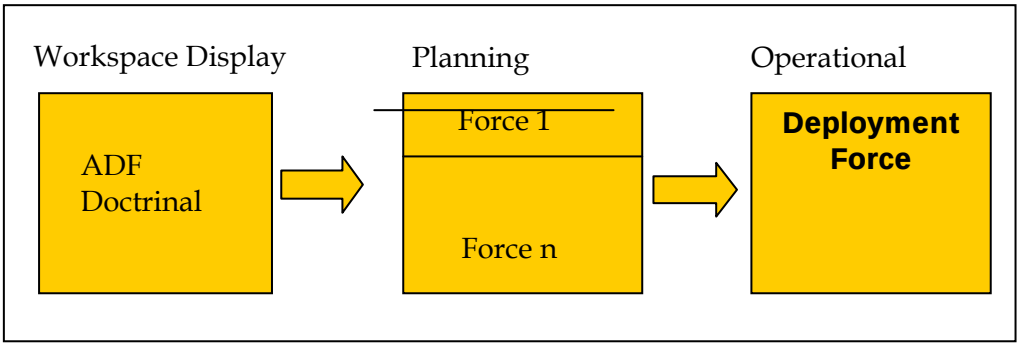


Figure 31 Workspace transformation from planning to operational

Operation 1: Create planning Workspace for Force Assembly

The LHQ planner would use their client to create a new empty Workspace that will contain ORBAT options for the operation to be conducted. This Workspace would be called OPTION X CONTINGENCY and by default would be created on the local LHQ ORBAT service. The metadata and constraints for the OPTION X CONTINGENCY Workspace is shown in Table 4 below.

Table 4: OPTION X CONTINGENCY Workspace metadata

Workspace Name:	OPTION X CONTINGENCY
Description:	Planning Workspace for Force Assembly
Derived From Template:	False
Is Template:	False
Master:	True
Intended Use:	Plan
Capability Constraint:	OLOC
ORBAT Hostility Allowed:	No constraints are required on this Workspace.

N.B. Access control may be used to restrict access to other planners at LHQ.

The planner creates two child Workspaces called BLUE FORCES and INTERNATIONAL CONTINGENT. The BLUE FORCES Workspace will contain friendly Workspaces and ORBATs. *Table 5* shows the metadata that would be used when creating the BLUE FORCES Workspace.

Table 5: BLUE FORCES Workspace metadata

Workspace Name:	BLUE FORCES
Description:	Contains all Workspace and ORBATs that are friends because of characteristics, behaviour or origin.
Derived From Template:	False
Is Template:	False
Master:	True
Intended Use:	Plan
Capability Constraint:	OLOC
ORBAT Hostility Allowed:	1000001 (Assumed Friend) 1000005 (Friend)

The INTERNATIONAL CONTINGENT Workspace would hold the Workspaces and ORBATs corresponding to the international parties involved. Table 6 shows the metadata that would be used to create the international Workspace.

Table 6: INTERNATIONAL CONTINGENT Workspace metadata.

Workspace Name:	INTERNATIONAL CONTINGENT
Description:	Contains all Workspace and ORBATs that represent the international parties involved.
Derived From Template:	False
Is Template:	False
Master:	True
Intended Use:	Plan
Capability Constraint:	OLOC
ORBAT Hostility Allowed:	1000001 (Assumed Friend) 1000005 (Friend)

The following diagram shows how the LHQ Planner interacts with the ORBAT tools to create the planning Workspace. Figure 33 shows the interaction in more detail.

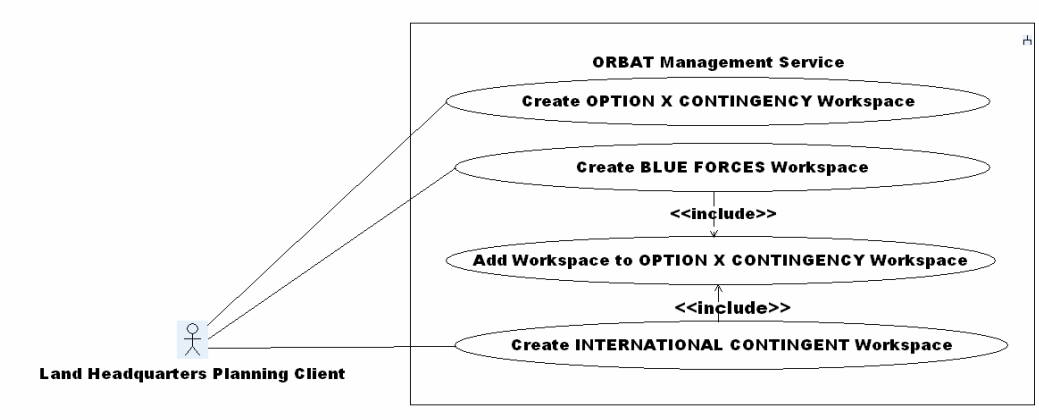


Figure 32 LHQ planner's interaction with ORBAT Services.

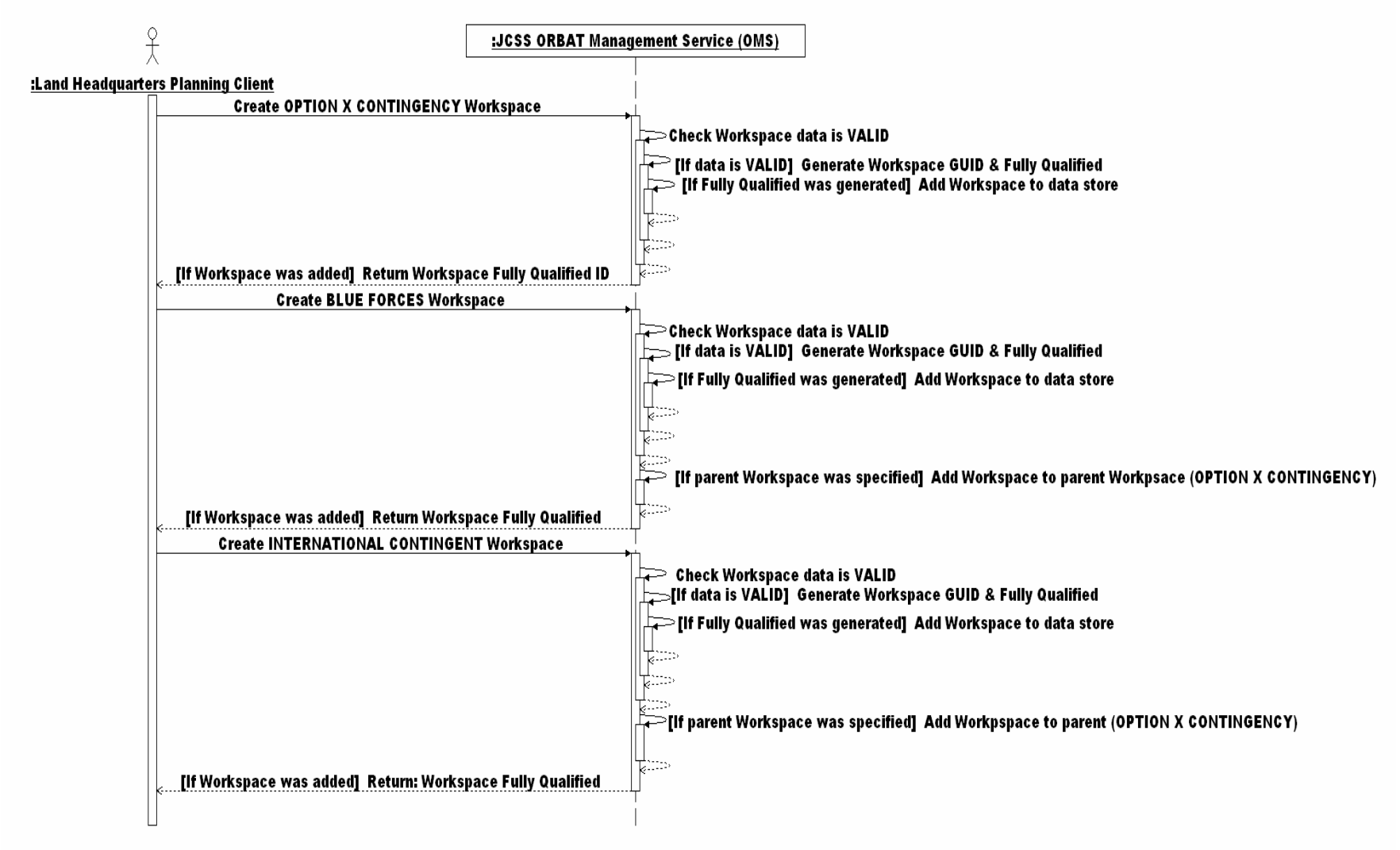


Figure 33 Process for completing operation 1.

Operation 2: Find ADF DOCTRINAL Workspace

The planner would use their client to search for the ADF DOCTRINAL Workspace. The client software would return the location and metadata of all known replicas of this Workspace.

The planner would select the most appropriate replica, based the location of the Workspace or its timeliness (e.g. CADTC may have a replica hosted on their service as well as MHQ, AHQ and CATDC). The planner's client software would retrieve the details of the ADF DOCTRINAL Workspace from the selected server (e.g. the CATDC version may be the master version).

The planner views the Workspace details, and retrieves the ARMY DOCTRINAL Workspace details. Table 7 shows the metadata associated with the Workspace. The ARMY DOCTRINAL Workspace may contain a number of ORBATs (how many ORBATs are used to represent the ARMY structure is up to the end users, it doesn't affect this use case). The Planner browses the ORBATs to check they have the right Workspace.

> N.B. This Workspace could also contain other ORBATs or Workspaces that contained future or historical forces, so the timeframe attribute of the ORBAT would be important here. This feature may be useful to track changes in doctrinal structure over time.

Table 7: ARMY DOCTRINAL Workspace metadata

Workspace Name:	ARMY DOCTRINAL
Description:	Contains the structure of the totality of all land forces potentially available.
Derived From Template:	False
Is Template:	False
Master:	True
Intended Use:	Doctrine
Capability Constraint:	OLOC
ORBAT Hostility Allowed:	1000001 (Assumed Friend) 1000005 (Friend)

Figure 34 shows the steps the LHQ planner would take to complete operation 2. Figure 35 show steps and the LHQ planners interaction with the ORBAT tool in more detail.

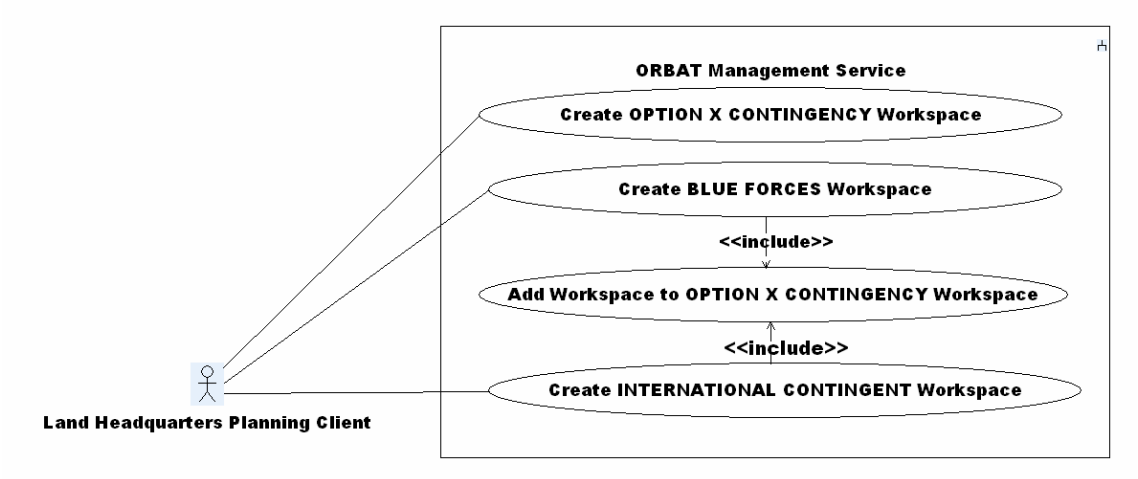


Figure 34 LHQ planner's interaction with ORBAT tools for operation 2

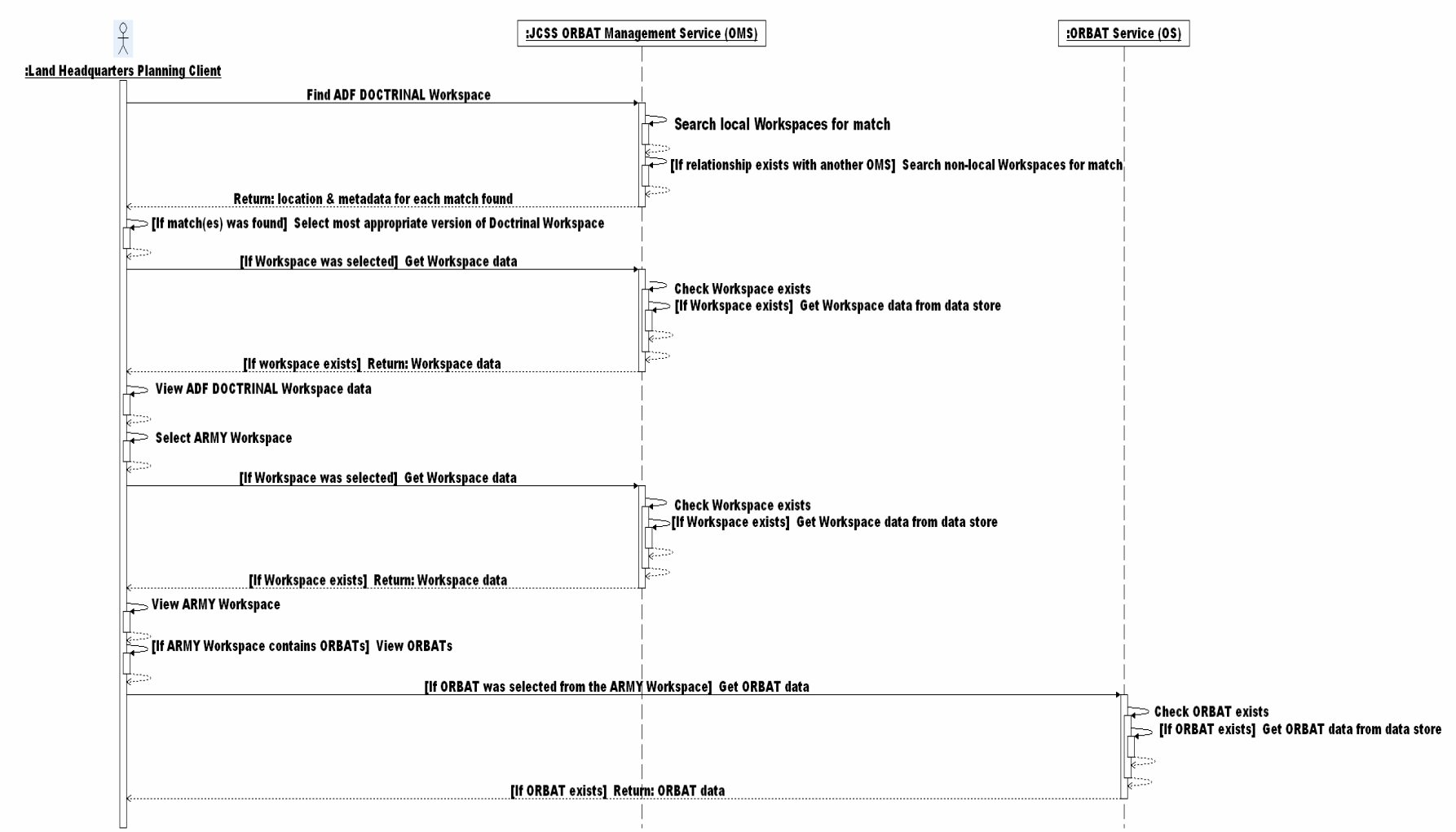


Figure 35 LHQ planners interaction with ORBAT tools

THIS PAGE IS INTENTIONALLY BLANK

Operation 3: Populate Planning Workspace for Force Assembly

Using the ORBATs available in the ARMY Workspace, the LHQ planner would select the force that is to be added to the BLUE FORCES Workspace. For this use case, the planner will select the 1 ARMD REGT ORBAT. Table 8 shows the metadata for this ORBAT.

Table 8: 1 ARMD REGT ORBAT metadata

ORBAT Name:	1 ARMD REGT
Exercise:	False
Derived From Template:	True (created from Armoured Regiment template)
Is Template:	False
Intended Use:	Doctrine
Authoritative:	True
Fictional:	False
Capability Constraint:	OLOC
Unit hostility allowed:	1000001 (Assumed Friend) 1000005 (Friend)

The client software would now be used to copy the selected ORBAT from the ARMY Workspace to the BLUE FORCES Workspace. The entire 1 ARMD REGT ORBAT will be copied; the planner can edit the ORBAT (eg: to delete units that are no longer required) at a later stage.

> N.B. If many options are being explored, a series of child Workspaces could be created and manipulated.

1 ARMD REGT ORBAT currently contains the following unit structure.

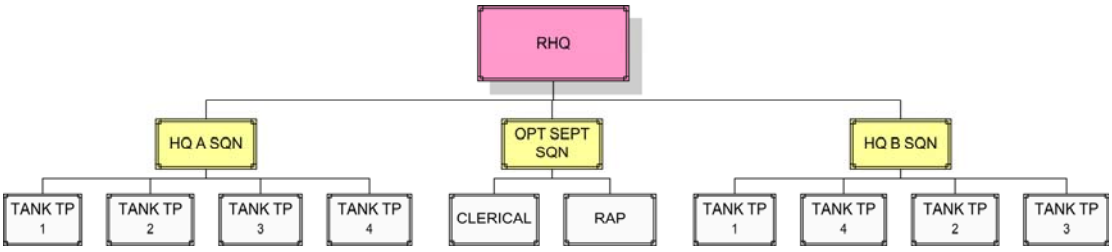


Figure 36 Current unit structure (hierarchy) of 1 ARMD REGT ORBAT

The force to be assembled requires a tank squadron, headquarters and a reconnaissance troop. The planner would use the client to remove the following units from the 1 ARMD REGT ORBAT in the planning Workspace.

- CLERICAL
- RAP
- HQ B SQN
- TANK TP 5
- TANK TP 6
- TANK TP 7
- TANK TP 8
- OPT SEPT SQN

The client will now be used to add a reconnaissance troop (RECON TP) to the ORBAT. The unit hierarchy is altered to add this unit to be under the command of the RHQ.

The final hierarchy is shown below.

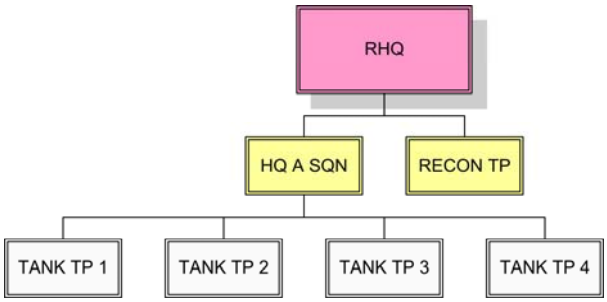


Figure 37 1 ARMD REGT ORBATs updated unit structure.

Table 9 below shows metadata for the units that are contained in the 1 ARMD REGT ORBAT after the hierarchy was updated.

Table 9: 1 ARMD REGT unit metadata

EID	*Name*	*Abbrev*
1234501150	Regional Headquarters	RHQ
1234503368	Headquarters A Squadron	HQ A SQN
1234507458	Tank Troop 1	TANK TP 1
1234506987	Tank Troop 2	TANK TP 2
1234512345	Tank Troop 3	TANK TP 3
1234509999	Tank Troop 4	TANK TP 4
1234501255	Reconnaissance Troop	RECON TP

N.B. The Enterprise Identifier Service has assigned the EIDs represented in the table above to the unit. Please see Chapter 7 for further details.

Now that the force has been assembled, the equipment and billet information can be edited. Below is an example of the XML that is held about a unit, its equipment and billets. Some optional elements and attributes have not been included in this set of data; please see Section 4.4.3 for more information about unit data.

N.B. The following data is fictional data that has been constructed especially for the use case.

```
<UNIT eid="1234501150" name="Regional Headquarter" abbrev="RHQ">

        <ALTERNATE_UNIT_ID>
            <CID_NUMBER>2</CID_NUMBER>
        </ALTERNATE_UNIT_ID>
        <xsdb:BILLET role="CO" vacant="false" billetCapability="oloc" number="1" rank="Col">
            <xsdb:POSITON_ID>
                <xsdb:ARMY_POSITION_NO id="748629"/>
            </xsdb:POSITON_ID>
        </xsdb:BILLET>
        <xsdb:BILLET role="OPS OFFR" vacant="false" billetCapability="oloc" number="2" rank="Maj">
            <xsdb:POSITON_ID>
                <xsdb:ARMY_POSITION_NO id="123456"/>
            </xsdb:POSITON_ID>
        </xsdb:BILLET>
        <xsdb:BILLET role="LO" vacant="false" billetCapability="oloc" number="5" rank="String">
            <xsdb:POSITON_ID>
                <xsdb:ARMY_POSITION_NO id="123456"/>
            </xsdb:POSITON_ID>
        </xsdb:BILLET>
        <xsde:UNIT_EQUIPMENT_HOLDINGS quantity="12" status="Fully Operational" equipmentCapability="oloc"
startDate="2003-08-13" endDate="2004-08-13" onLoan="true">
            <xsde:EQUIPMENT_ID>
                <xsde:SIGC_EGC idNumber="9999-7771" equipmentDesignation="PISTOL 9MM AUTO"/>
        </xsde:UNIT_EQUIPMENT_HOLDINGS>
        <xsde:UNIT_EQUIPMENT_HOLDINGS quantity="2" status="Fully Operational" equipmentCapability="oloc"
startDate="2003-08-14" endDate="2003-12-13" onLoan="true">
            <xsde:EQUIPMENT_ID>
                <xsde:SIGC_EGC idNumber="1234-1234" equipmentDesignation="CARBINE 5.56MM AUSTEYR
F88-C"/>
            </xsde:EQUIPMENT_ID>
        </xsde:UNIT_EQUIPMENT_HOLDINGS>
        <xsde:UNIT_EQUIPMENT_HOLDINGS quantity="50" status="Fully Operational" equipmentCapability="oloc"
onLoan="false">
            <xsde:EQUIPMENT_ID>
                <xsde:SIGC_EGC idNumber="5001-3700" equipmentDesignation="RIFLE 5.56MM AUSTEYR F88-
S"/>
            </xsde:EQUIPMENT_ID>
        </xsde:UNIT_EQUIPMENT_HOLDINGS>
        <xsdts:UNIT_TIMESTAMP>
            <xsdts:DATE_FIRST_CREATED>2003-12-17T09:30:47-05:00</xsdts:DATE_FIRST_CREATED>
            <xsdts:DATE_LAST_UPDATED>2003-12-20T10:59:47-05:00</xsdts:DATE_LAST_UPDATED>
        </xsdts:UNIT_TIMESTAMP>
        <UNIT_DESCRIPTION unit-type-size-code="1000012" unit_hostility_code="1000005"/>
    </UNIT>
```

Figure 38 XML representation of the unit data held by an ORBAT Service

More ORBATs are now created to store the rest of the planning data. They can be uploaded to the local ORBAT service (with some access control preventing them being confused with more-developed planning). They could also be exported to an XML document, and emailed directly to some other planning process. Obviously, for change control reasons, uploading information to a server is always preferable.

N.B. Assuming that a source of personnel and equipment data was available for the units selected (say in a UNIT PLOC Workspace, located with the actual unit), the tool could use the linking information in the ORBAT to generate logistics (movement, sustainment), health, and finance information. This would be akin to checking in with the PLOC Workspace to get the *actual* figures instead of relying on establishment figures that are usually associated with Doctrinal ORBATS. This may be a good idea given that some units are significantly different from their doctrinal representation.

The planner would use the client to create an INTERNATIONAL SECURITY FORCE ORBAT for the INTERNATIONAL CONTINGENT Workspace. The INTERNATIONAL SECURITY FORCE ORBAT will contain information regarding the security personnel and equipment that is from an international source.

NB: At the time of creation, the INTERNATIONAL SECURITY FORCE ORBAT will be empty. When we become aware of the contributions being made by other nations the ORBAT can be populated.

Table 10: INTERNATIONAL SECURITY FORCE ORBAT metadata

ORBAT Name:	INTERNATIONAL SECURITY FORCE
Exercise:	False
Derived From Template:	False
Is Template:	False
Intended Use:	Operational
Authoritative:	True
Fictional:	False
Capability Constraint:	OLOC
Unit hostility allowed:	1000001 (Assumed Friend) 1000005 (Friend)

Figure 39 below shows the ORBAT tool the LHQ planner interacts with to complete this operation. Figure 40, Figure 41 and Figure 42 show the LHQ planner's interaction with the system in more detail.

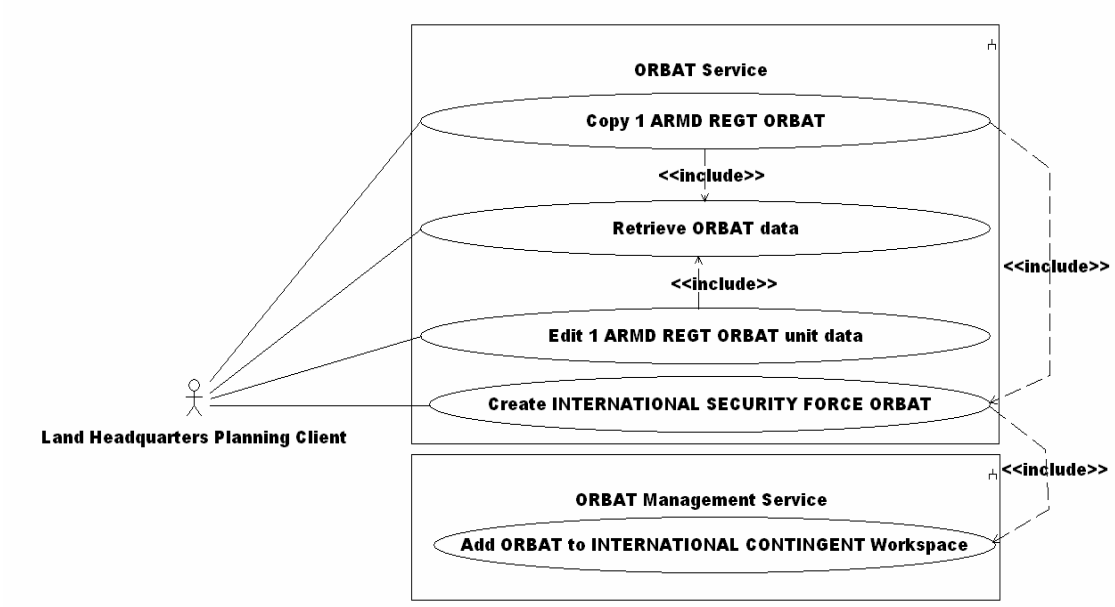


Figure 39 LHQ planner's interaction with ORBAT tools for operation 3.

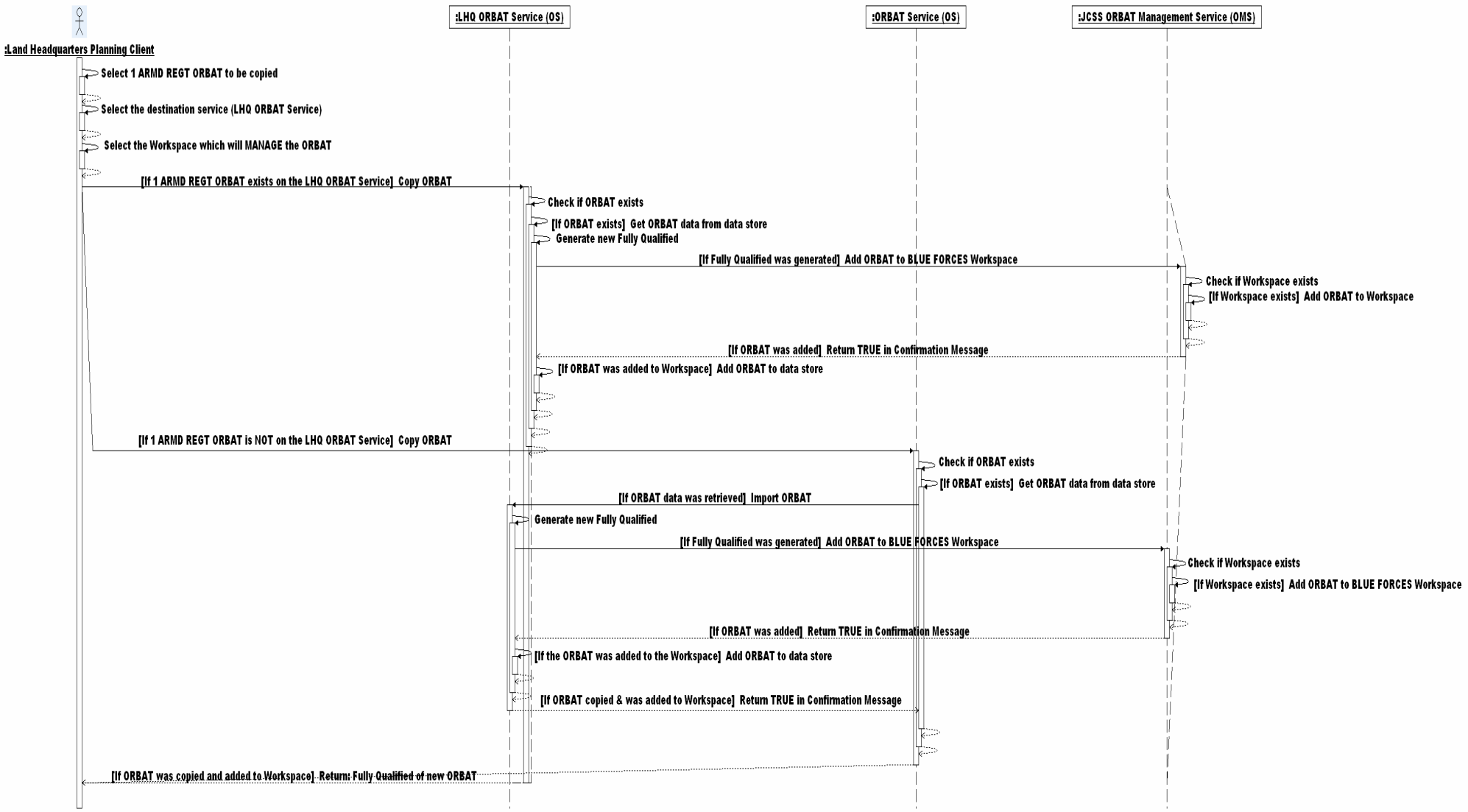


Figure 40 Series of events for copying 1 the ARMD REGT ORBAT.

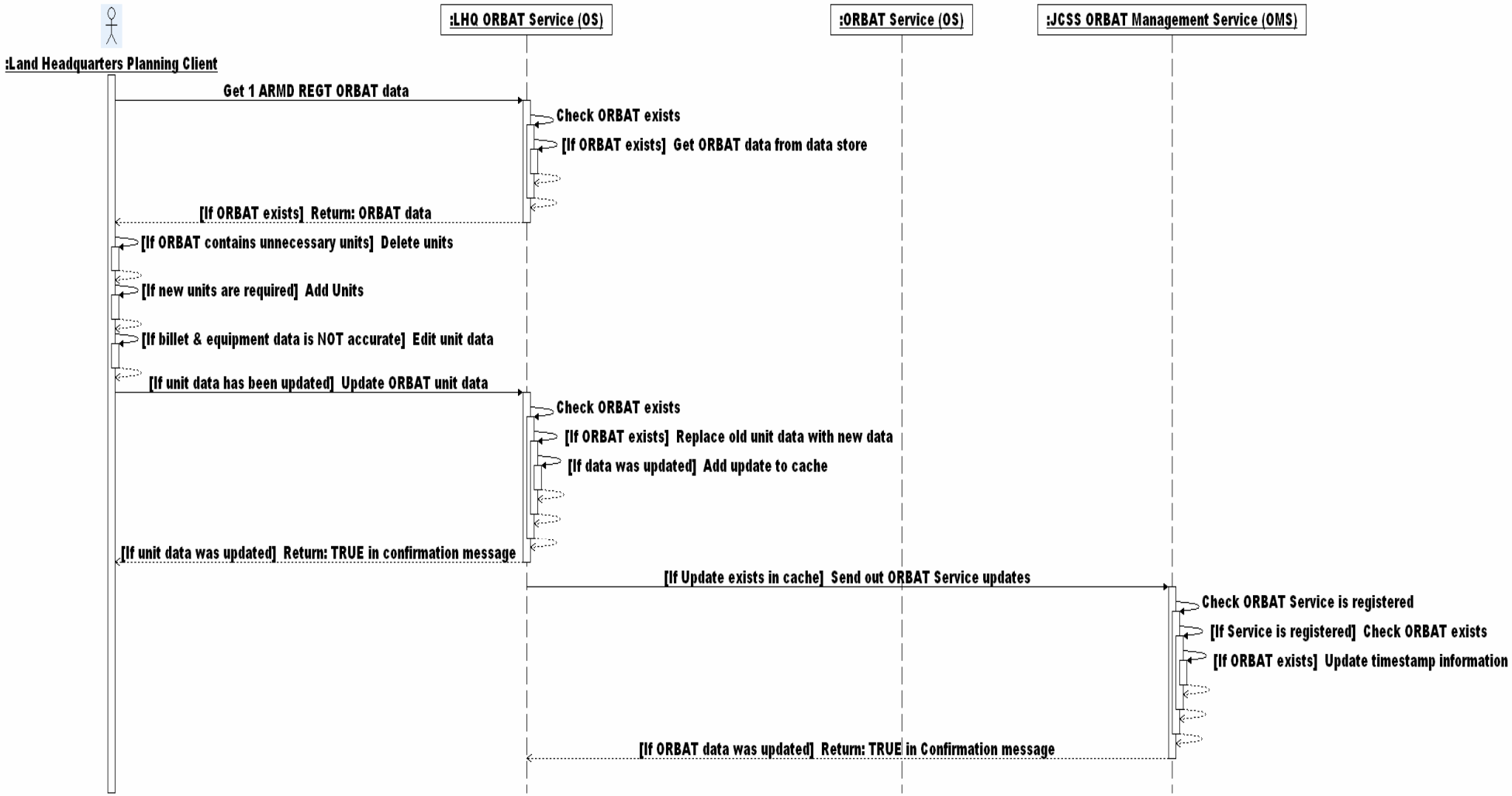


Figure 41 Series of events for editing 1 ARMD REGT ORBAT data.

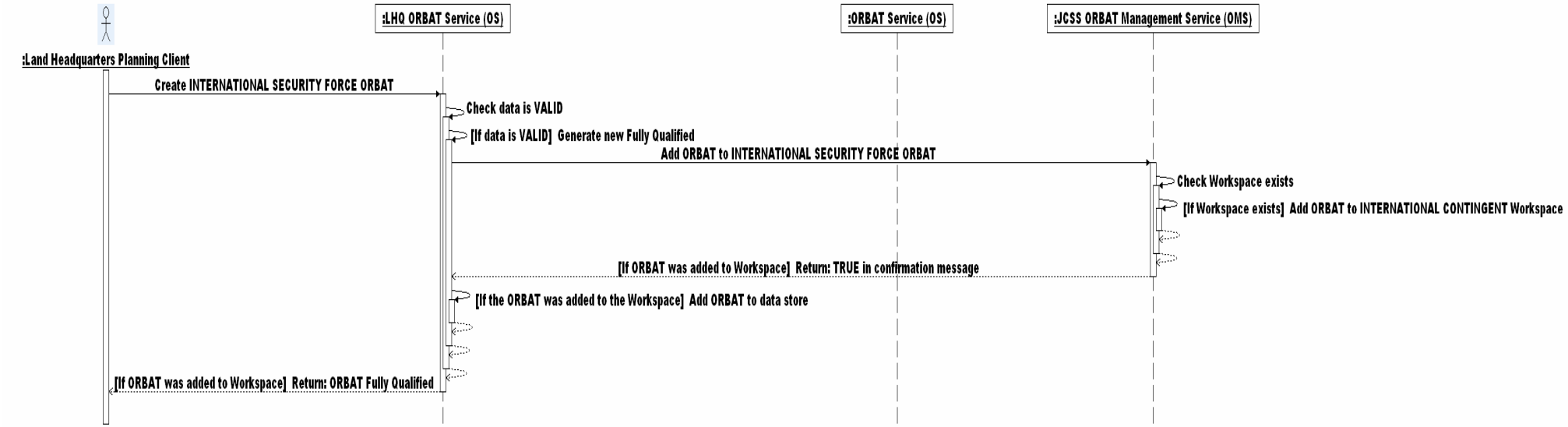


Figure 42 Series of events for creating the INTERNATIONAL SECURITY FORCE ORBAT

Operation 4: Create Operational Workspace

Once the normal processes have been applied and a planning option has been approved, it is possible to create an operational Workspace. Creating the operational Workspace may be as simple as placing the preferred planning option in its own Workspace. The client would be used to copy the OPTION X CONTINGENCY Workspace, the planner would then rename the Workspace to something like OPERATION X. The metadata for the Workspace will also change; the intended use attribute will be changed from plan to operational, other metadata may also need changing. Table 11 shows the metadata of the operational Workspace.

Security permissions would most likely be changed at this point, giving broader visibility to the Workspace. Access control would also be important when senior staff are transitioning ORBATs from the Force Assembly Workspace into the Deployment Force ORBAT/Workspace.

Table 11 OPERATION X Workspace metadata

Workspace Name:	OPERATION X
Description:	Operational Workspace created from planning Workspace.
Derived From Template:	False
Is Template:	False
Master:	True
Intended Use:	Operational
Capability Constraint:	OLOC
ORBAT Hostility Allowed:	No constraints are required on this Workspace.

Figure 43 shows how the planner would place the preferred planning option in its own Workspace to create an operational Workspace. Figure 44 below shows the steps involved in creating an operational Workspace in more detail.

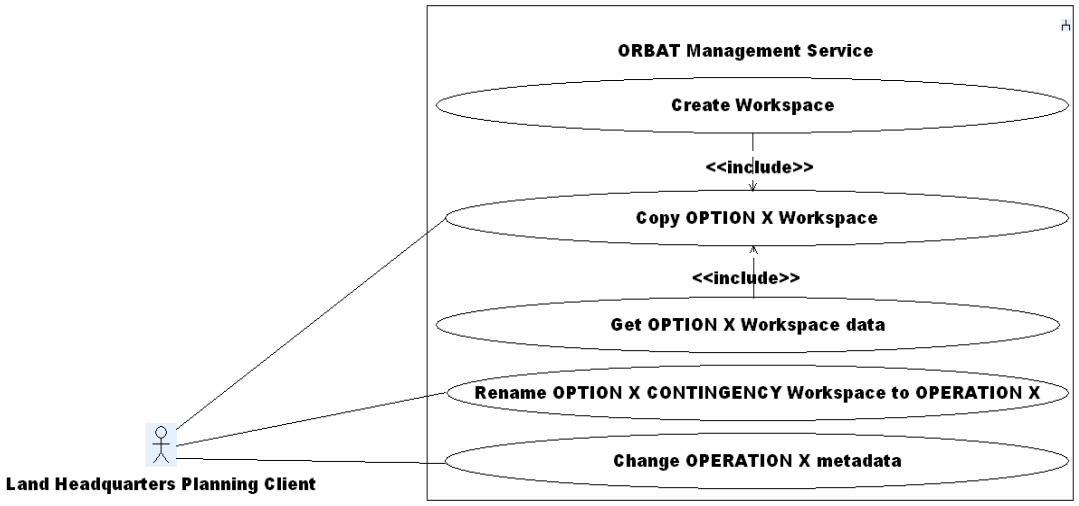


Figure 43 LHQ planner using the ORBAT Tools to create an operational Workspace

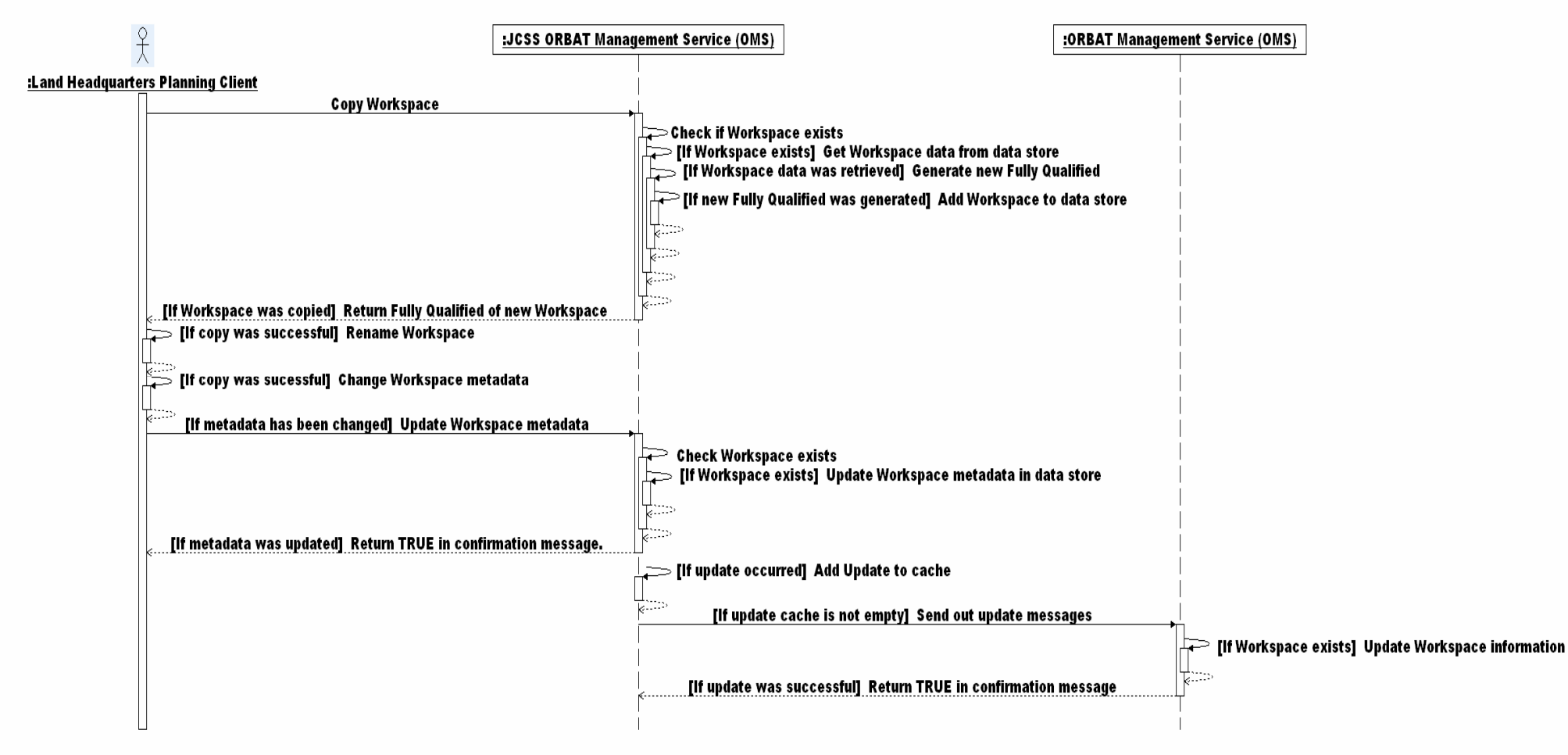


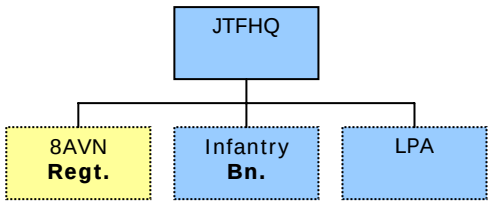
Figure 44 Sequence of events for creating an operational Workspace.

8.2 MRO related ORBAT Service Usage

This use case explores how a MRO may be represented and used.

MRO 1234
SPE of Country X

Requires an LPA, an infantry battalion, and 8th Aviation Regiment, specifically. This might be due to their unique equipment, and skills.



Light blue units in the ORBAT are 'templates', pointing to capabilities, rather than specific units. The yellow unit is a specific unit, which must be available if the MRO is to be activated. The dashed lines are 'pointers', or references to units contained in other ORBATs; they represent leaves in the MRO ORBAT. When the MRO plan moves into an operation (named ***OP Whiteboard***) they will be expanded into a full ORBAT. The main JTF HQ 'unit' in the ORBAT would be a full structure (detailed down to billets), but the billets wouldn't point to real people as this information would be held in another Workspace.

Operations to be tested:

1. Check that PLOC for ***8AVN*** meets the required MLOC for this MRO
2. Find all infantry battalions that meet MLOC for this MRO
3. Creating an operation ORBAT from the MRO
4. How many people are there in the ***OP Whiteboard*** JTF
5. What materiel is to be moved for the ***OP Whiteboard*** JTF

Operation 1: Check that PLOC for 8AVN meets the required MLOC for this MRO

The client software queries each ORBAT Service and requests the location of all ORBATs that hold PLOC information about the 8th Aviation Regiment (8AVN).

The query returns references to three services that hold a PLOC ORBAT containing information about this unit: LHQ, ADFWC, and 8AVN itself. In normal operations only one service is allowed to be the 'authoritative' source for PLOC data, in this case it is the 8AVN ORBAT service itself. (See the Information Management Use Case for further information on how this is managed.) The information returned for each ORBAT matched contains the ORBAT metadata, timestamps and fully qualified id. The series of tables below show the metadata returned for each match, along with the last updated timestamp for the entire ORBAT.

N.B. Each different area of an ORBAT's data encompasses its own timestamps. For example, both the unit data and metadata contain individual last updated timestamps. The timestamp shown in the table is the ORBAT timestamp, indicating the last time any aspect of the ORBAT was updated.

Table 12: 8AVN ORBAT Service ARMY_PLOC ORBAT metadata & timestamp

Service Name:	8 AVN ORBAT Service
ORBAT Name:	ARMY_PLOC
Exercise:	False
Derived From Template:	False
Is Template:	False
Intended Use:	Doctrine
Authoritative:	True
Fictional:	False
Capability Constraint:	PLOC
Unit hostility allowed:	1000001 (Assumed Friend) 1000005 (Friend)
ORBAT last updated:	13/04/2003

Table 13 HQAST ORBAT Service ARMY_PLOC ORBAT metadata & timestamp

Service Name:	HQAST ORBAT Service
ORBAT Name:	ARMY_PLOC
Exercise:	False
Derived From Template:	False
Is Template:	False
Intended Use:	Doctrine
Authoritative:	False
Fictional:	False
Capability Constraint:	PLOC
Unit hostility allowed:	1000001 (Assumed Friend) 1000005 (Friend)
ORBAT last updated:	18/04/2003

Table 14 LHQ ORBAT Service ARMY_PLOC ORBAT metadata & timestamp

Service Name:	LHQ ORBAT Service
ORBAT Name:	ARMY_PLOC
Exercise:	False
Derived From Template:	False
Is Template:	False
Intended Use:	Doctrine
Authoritative:	False
Fictional:	False
Capability Constraint:	PLOC
Unit hostility allowed:	1000001 (Assumed Friend) 1000005 (Friend)
ORBAT last updated:	13/04/2003

When the decision is being made about which ORBAT to use, the current status of the service where the ORBAT is located needs to be taken into consideration. Table 15 below shows the current status of the three services that contain the ORBATs we are interested in.

Table 15 current statuses of ORBAT Services

ORBAT Service Name:	*Status*
8AVN	Deployed – Not Available
HQAST	Available
LHQ	Available

N.B. The current status of each ORBAT Service is not returned with the results from the query performed earlier; the status needs to be checked separately.

The client software looks at the results and determines that HQAST version is most appropriate. The rationale for this decision is that it is more recent than the LHQ version and the Authoritative version (sitting on 8AVN ORBAT Service) is not available due to a current deployment.

N.B. This choice could be coded into the client software or made by the user.

Using a similar process, the MLOC ORBAT for this MRO is retrieved from the ORBAT service running at DG Preparedness. A business process[1] is then applied to decide whether or not the PLOC ORBAT matches the MLOC ORBAT.

Figure 45 shows the steps that need to be completed in order to check that PLOC for 8AVN meets the required MLOC for this MRO. Figure 46 shows these steps in more detail.

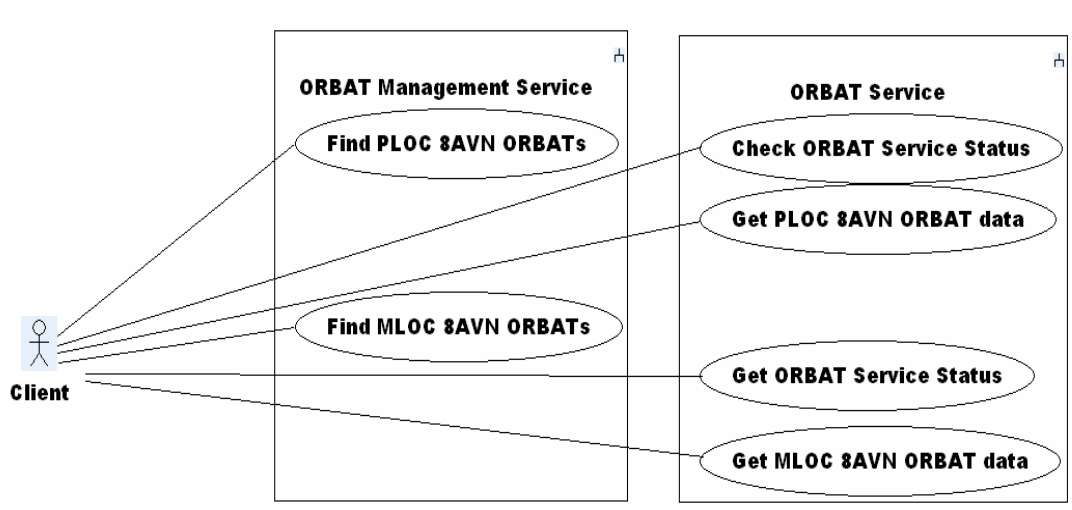


Figure 45 Interaction with ORBAT tools to complete operation 1.

[1] Perhaps this could look at the number of pilots and helicopters available in the PLOC versus the numbers in the MLOC – it is not a concern of the ORBAT service itself. This good example of a specialist ORBAT tool using the data.

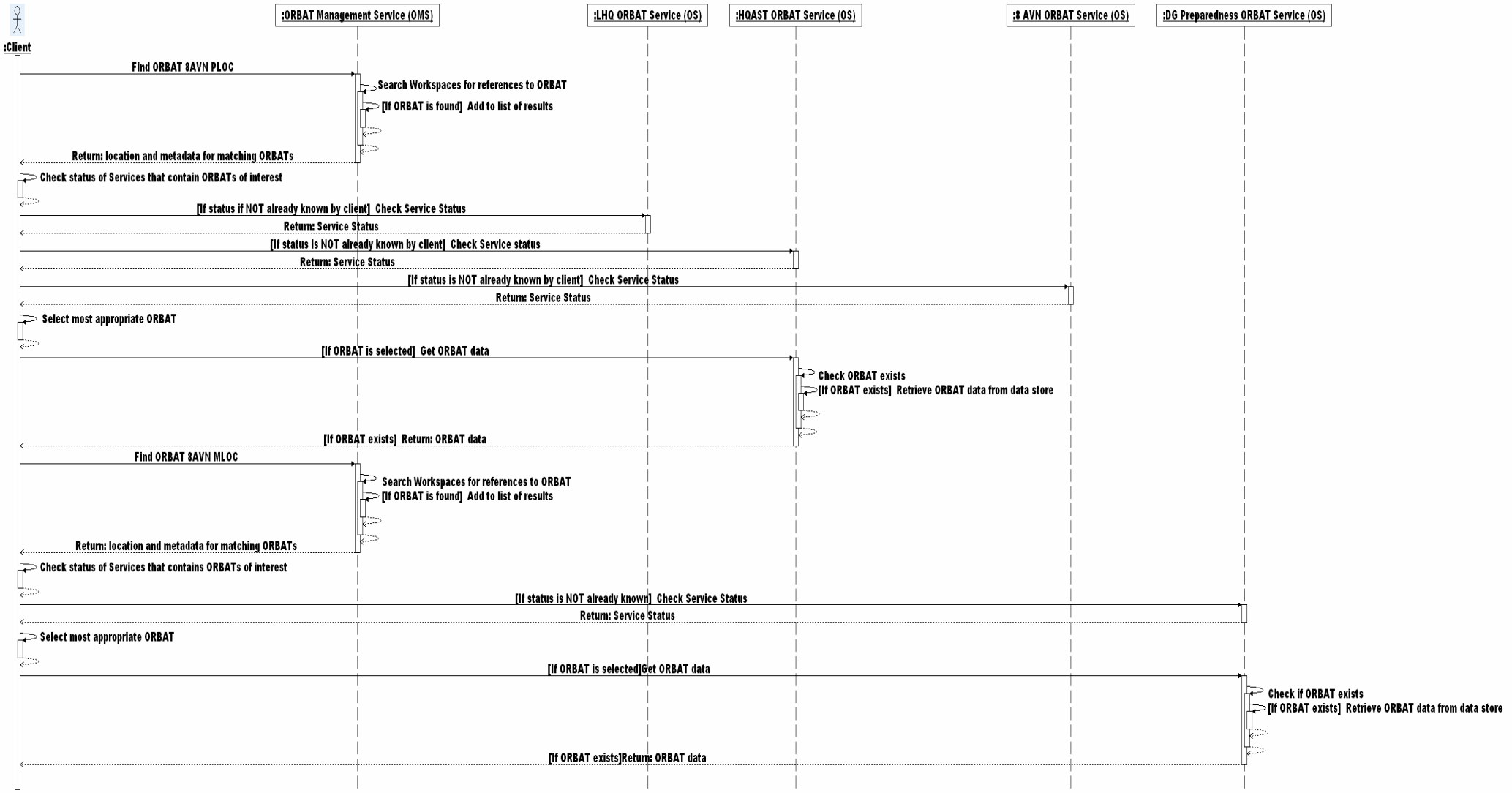


Figure 46 Interactions with different ORBAT tools to complete operation 1.

Operation 2: Find all infantry battalions that meet MLOC for this MRO

The client performs a search on the ORBAT Management Services to locate the authoritative version of the ADF PLOC Workspace and retrieves its data. The metadata for the ADF PLOC Workspace is shown in Table 16, below.

N.B. In the event that the authoritative ADF PLOC Workspace is unavailable, a decision would be made about which available version would be the most appropriate to use instead.

Table 16: ADF PLOC Workspace metadata

Workspace Name:	ADF PLOC
Description:	Contains the structure of the totality of all Australian Defence Forces potentially available. Represents their present level of capability.
Derived From Template:	False
Is Template:	False
Master:	True
Intended Use:	Operational
Capability Constraint:	OLOC
ORBAT Hostility Allowed:	100001 (Assumed Friend) 100005 (Friend)

The client software would request the master ADF PLOC Workspace from the ORBAT Management Service where it is located. A search would then be performed to locate any ORBATs that are derived from the template for a doctrinal Infantry Battalion ORBAT.

> N.B. When an ORBAT is created to represent a specific structure (i.e. an Infantry Battalion) the template ORBAT for that specific structure will be copied and then populated with the relevant information. Within the new Infantry Battalion ORBAT data, the fact it was created from a template will be recorded.

For each of the Infantry Battalion ORBATs, the PLOC could be checked against the battalion MLOC for the MRO.

Assuming that more than one Battalion fits the bill, the Current Operations Workspace would be checked to see which of the candidate battalions was already on operations, or flagged for operational duty during the time of our planned operation[2].

After comparing the MLOC and PLOC data for the MRO, and checking the Current Operations Workspace, it is decided that Australia's 1st Infantry Battalion is best suited to the Operation. Table 17 shows the 1st Infantry Battalion ORBAT's metadata.

[2] Obviously rotational cycles, etc, have to be considered as well; this is merely one check that can be partly automated.

Table 17: 1st INFANTRY BATTALION ORBAT metadata

ORBAT Name:	1st INFANTRY BATTALION
Exercise:	False
Derived From Template:	TRUE
Derived From Template id:	1234501799
Derived From Template name:	ADF INFANTRY BATTALION
Is Template:	False
Intended Use:	Plan
Authoritative:	True
Fictional:	False
Capability Constraint:	PLOC
Unit hostility allowed:	1000001 (Assumed Friend) 1000005 (Friend)

Figure 47 shows the functions that are invoked on the ORBAT tools in order to complete this operation. Figure 48 shows the invocation of the methods in more detail.

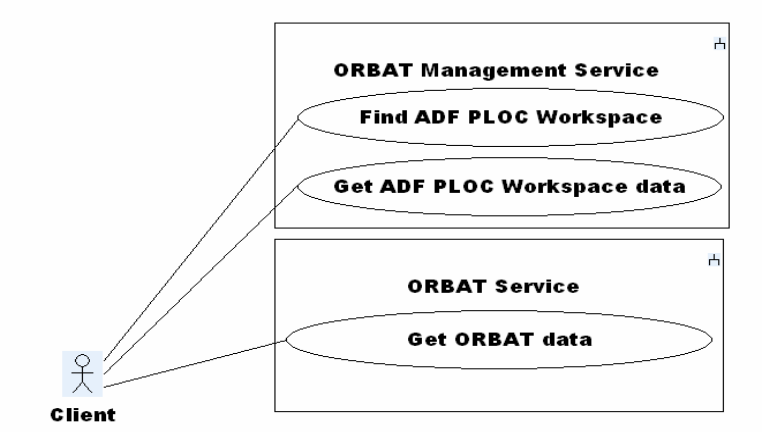


Figure 47 Interaction with ORBAT tools for operation 2

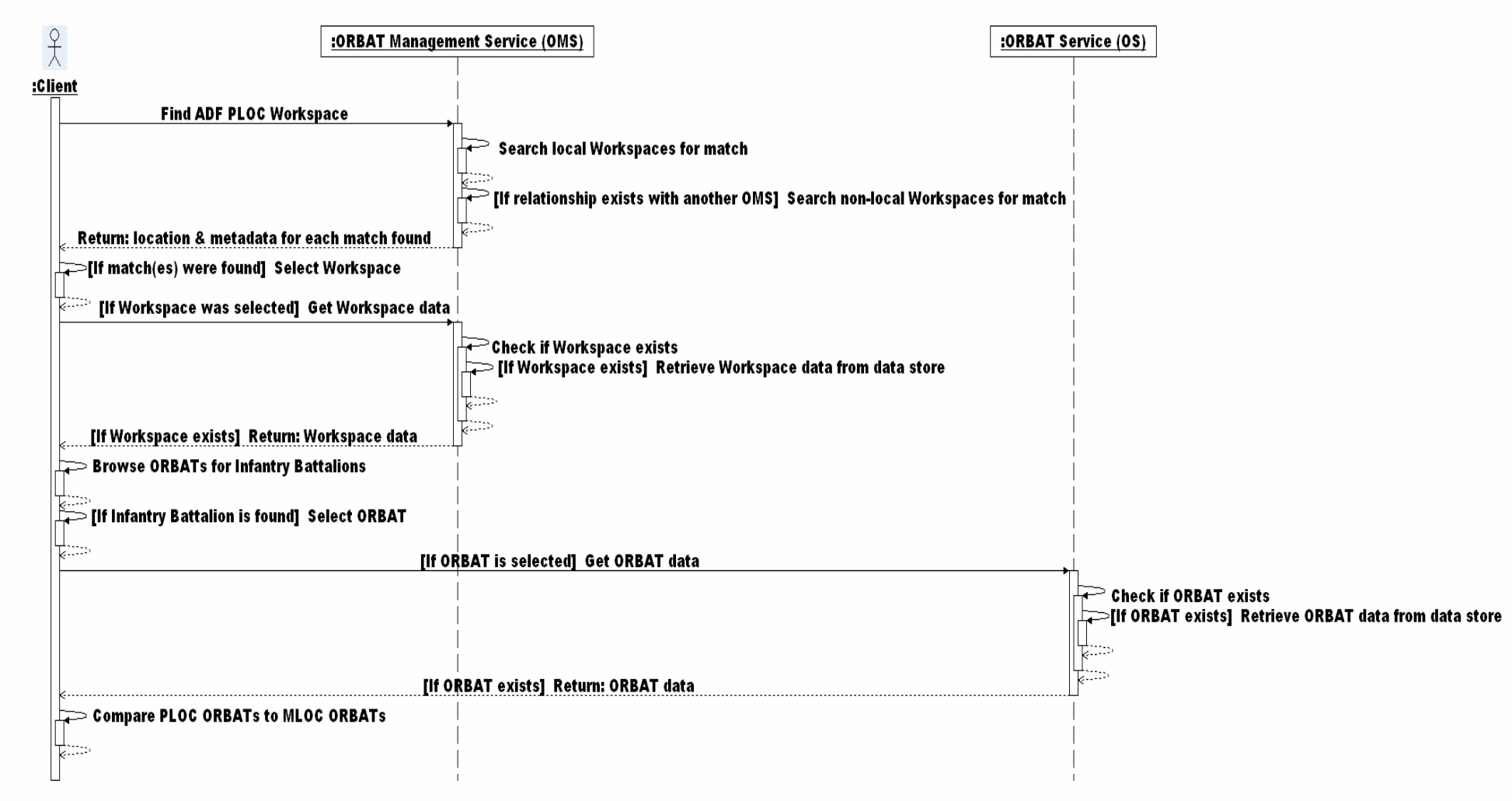


Figure 48 Steps involved in completing operation 2

Operation 3: Creating an operational ORBAT from the MRO

An operational Workspace is created (named OP Whiteboard JTF) using the client. This Workspace will contain the operational ORBATs for MRO 1234. Table 18 below shows the metadata for the operational Workspace.

Table 18 OP WHITEBOARD JTF Workspace metadata

Workspace Name:	OP WHITEBOARD JFT
Description:	Contains the Operational ORBATs for MRO 1234: SPE of Country X
Derived From Template:	False
Is Template:	False
Master:	True
Intended Use:	Operational
Capability Constraint:	OLOC
ORBAT Hostility Allowed:	100001 (Assumed Friend) 100005 (Friend)

Using the client software, the 1st INFANTRY BATTALION ORBAT is copied into the OP WHITEBOARD JTF Workspace. The metadata for the ORBAT has to be changed to reflect its new operational context. I.e.: The intended use attribute within the OP WHITEBOARD JTF Workspace is now Operational, whereas previously it was used for planning purposes. The modified metadata is shown in Table 19 below.

Table 19 1st INFANTRY BATTALION ORBAT modified metadata

ORBAT Name:	1st INFANTRY BATTALION
Exercise:	False
Derived From Template:	TRUE
Derived From Template id:	1234501799
Derived From Template name:	ADF INFANTRY BATTALION
Is Template:	False
Intended Use:	Operational
Authoritative:	True
Fictional:	False
Capability Constraint:	PLOC
Unit hostility allowed:	1000001 (Assumed Friend) 1000005 (Friend)

As this operational ORBAT was created from a template, the references to types of units need to be replaced by references to real units (i.e. the ORBAT might specify that the battalion is made up of two field batteries, the actual data and names for these two field batteries has not been supplied, just the structure).

When this ORBAT is 'saved' to an ORBAT service, business logic would be used to check that the references aren't pointing to template units, and that the operator submitting the ORBAT has the authorisation to do so.

Figure 49 shows what needs to be done to create an operational ORBAT from an MRO using the ORBAT tools. Figure 50 shows how the tools are used in more detail.

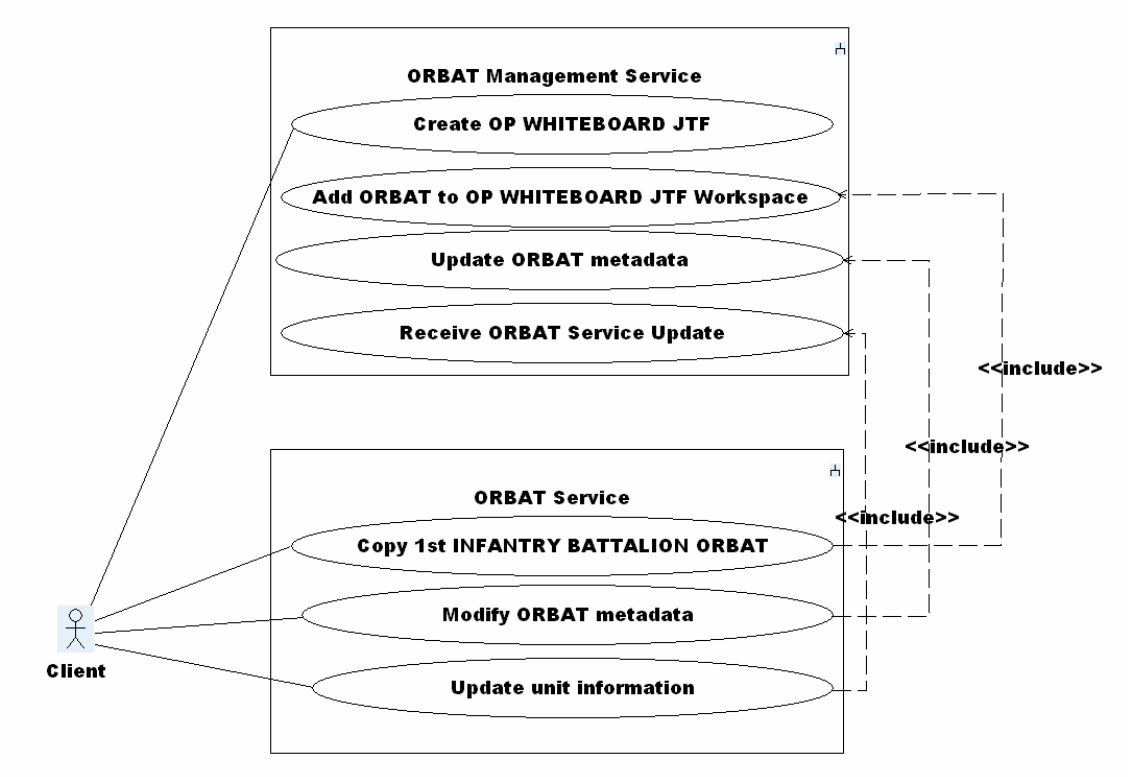


Figure 49 Invocation of methods on ORBAT tools

Figure 50 Creating an operations ORBAT from an MRO

Operation 4: How many people are there in the OP Whiteboard JTF

The client software traverses the operational Workspace and counts up the number of billets that actually point to real people, as opposed to 'empty' billets (positions to be filled, etc). The billets are specified at the ORBAT level, not at the higher Workspace level. The billets in each individual ORBAT contained within the OP WHITEBOARD JTF would have to be counted to get the total number.

If personnel information was available from PMKeys, etc., this could be cross-referenced to generate value-added information, such as breaking down the JTF by blood group for health planning purposes.

N.B. This functionality is client specific and as such ORBAT Management Services and ORBAT Services are not capable of performing such functions.

Operation 5: What materiel is to be moved for the OP Whiteboard JTF

The client software traverses the operational Workspace, and counts up the equipment, and quantities of equipment from all the units in the ORBATs contained within the Workspace. If a cataloguing service were available (based on surfacing data contained in DEFCAT, or its replacement CODEX) this could be cross-referenced from the linking information in the ORBATs, and used to create a highly detailed movement manifest.

N.B. This functionality is client specific and as such ORBAT Management Services and ORBAT Services are not capable of performing such functions.

8.3 Information Management

This use case describes how ORBAT Services (OS) interact with ORBAT Management Services (OMS) to provide consistent information management of Workspaces and ORBATs in a distributed environment. It focuses on the process that a user charged with preparing for and returning from a deployment would follow.

Scenario:

The DJFHQ has been deployed on disaster relief operations; the headquarters has been relocated to a remote locality and has established standard JCSS communications. 8AVN are in barracks and have been given notice to move under the DJFHQ at the remote location.

Preconditions:

1. There are three ORBAT services in this use case:
    a. 8AVN_Fixed
    b. 8AVN_Deployed
    c. DJFHQ_Deployed

2. There are three ORBAT Management Services:
    a. JCSS_Fixed
    b. 8AVN_Deployed
    c. DJFHQ_Deployed

Each ORBAT Service has a relationship with a single ORBAT Management Service. The OMS handles all complex queries and indexing issues on behalf of its registered ORBAT Service(s). At the start of this use case the following registrations have already occurred:

Table 20 Existing ORBAT Service & OMS relationships

ORBAT Service	*ORBAT Management Service*
8AVN_Fixed	JCSS_Fixed
8AVN_Deployed	8AVN_Deployed

Figure 51 shows the relationship between the ORBAT Services and which ORBAT Management Service they are registered to.

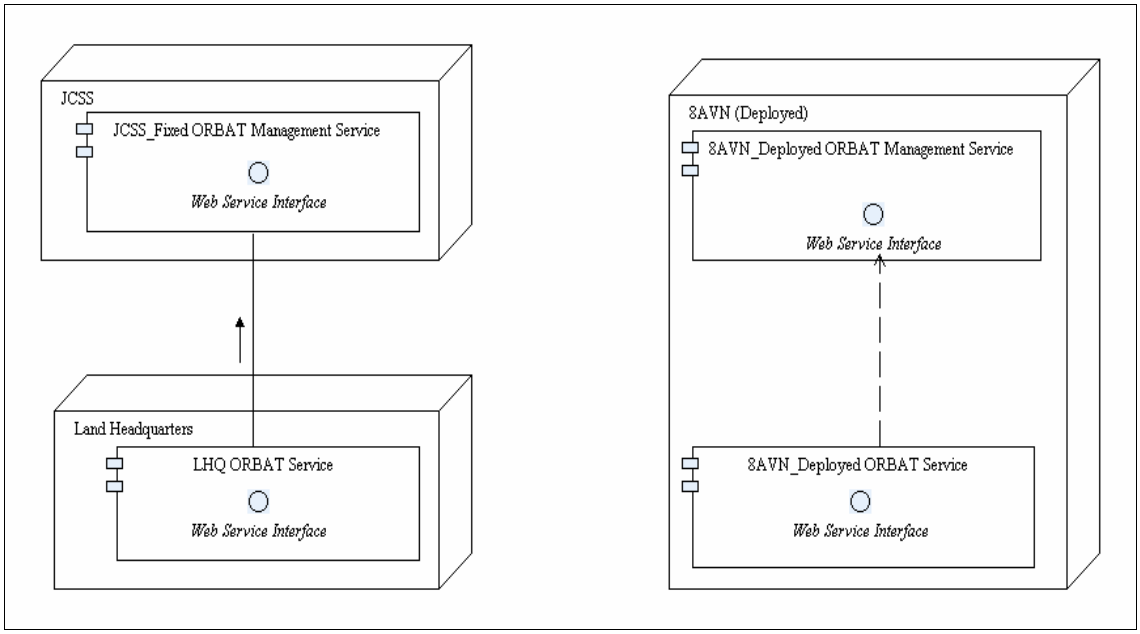


Figure 51 Current registrations of ORBAT Services

N.B. Each OMS can and probably will have multiple ORBAT services registered with it. For example, the JCSS_Fixed ORBAT Management Service could be the central OMS for the whole of the fixed JCSS network, thus the HQAST ORBAT Service would be registered to it. We have omitted these registrations for clarity, as they are not used in this example.

ORBAT Management Services replicate indexing information between each other in order to allow for efficient querying and robustness of the system. ORBAT Management Services do not contain the ORBAT data, they contain only the metadata index that identifies and is used to search for the correct instance of the data. Further details on the specifics of the OMS are contained in the service description in Section 6.

For this scenario the following OMS permission settings and index forwarding settings have been made:

Table 21 ORBAT Management Service permission settings

ORBAT Management Service	*Permission Settings*[3]	*Index Forwarding Options*
JCSS_Fixed	Allow all OMS connections	Act as forwarding agent
DJFHQ_Deployed	Allow JCSS Allow all deployed (will probably need a list) Deny others	Act as forwarding agent
8AVN_Deployed	Allow JCSS_Fixed Deny others	No forwarding

[3] This is a description only, the actual syntax used or combination of settings will differ.

Figure 52 shows the relationships that exist between the different ORBAT Management Services after the above settings have been implemented.

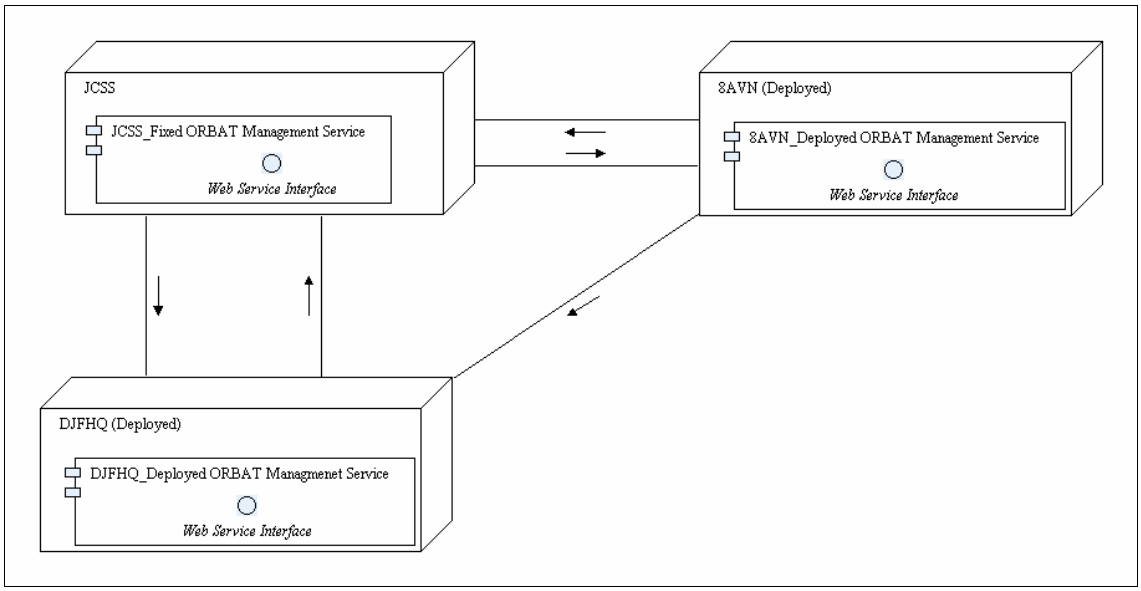


Figure 52 ORBAT Management Service relationships

The following replication relationships have been established between each OMS, replication entries are omnidirectional, i.e. one line per direction:

Table 22: Replication relationships

Source OMS	*Target OMS*	*Indexes Replicated*
8AVN_Deployed JCSS_Fixed DJFHQ_Deployed	JCCS_Fixed 8AVN_Deployed JCCS Fixed	All local Workspaces All local Workspace All local and non-local Workspaces

There will be three distinct phases of this use case; each phase consists of a number of operations to be tested:

Phase 1: Preparation for deployment

Operation 1: View contents of 8AVN_Deployed ORBAT Services

The local 8AVN data manager receives the notice to move. The data manager would use the OMS Client to query the 8AVN_Deployed ORBAT Management Service for its current Workspaces. A list of the current Workspaces is returned.

> N.B. All the Workspaces that the ORBAT Management Service knows about will be returned. It is the client's responsibility to provide a filtering function for showing local and non-local Workspaces.

Figure 53 shows the 8AVN data mangers interaction with the ORBAT tool. Figure 54 shows this interaction in more detail.

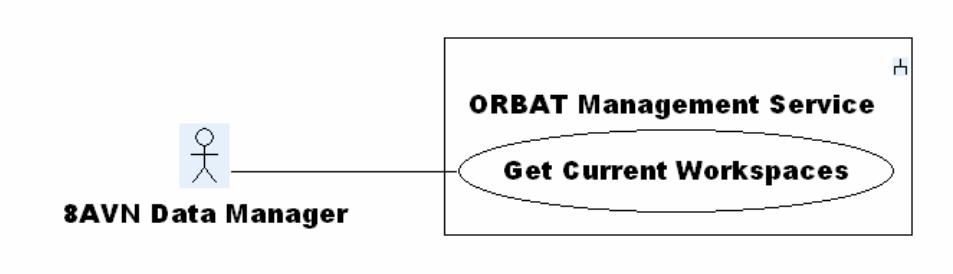


Figure 53 8AVN data manager's interaction with ORBAT tools

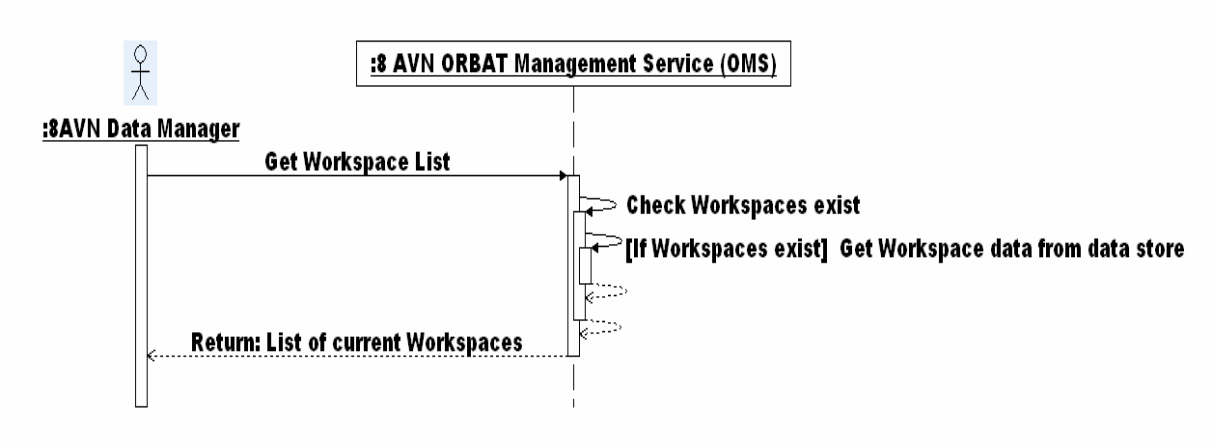


Figure 54 Sequence of events for operation 1

Operation 2: Delete unnecessary ORBATs and Workspaces

The data manager notices that several of the Workspaces and ORBATs returned are not relevant to this operation and should have been deleted at the end of the last exercise. The OMS Client would be used to determine if these ORBATs and Workspaces exist on the local services. It is determined that they exist on the 8AVN_fixed services and all replicas or non-authoritative Workspaces and ORBATs should be removed. The data manager proceeds to use the OMS Client to *remove* identified Workspaces and ORBATs.

N.B. Only replicas can be *removed* from services. The *delete* function must be invoked to remove authoritative or master Workspaces.

The OMS Client returns an error message indicating that one of the Workspaces to be *removed* is currently marked as the master version. The data manager realises their mistake and chooses to keep that Workspace. All the others are successfully removed.

Figure 55 shows how the 8AVN data manager would use the ORBAT tools to complete operation 2. Figure 56 shows the interaction in more detail.

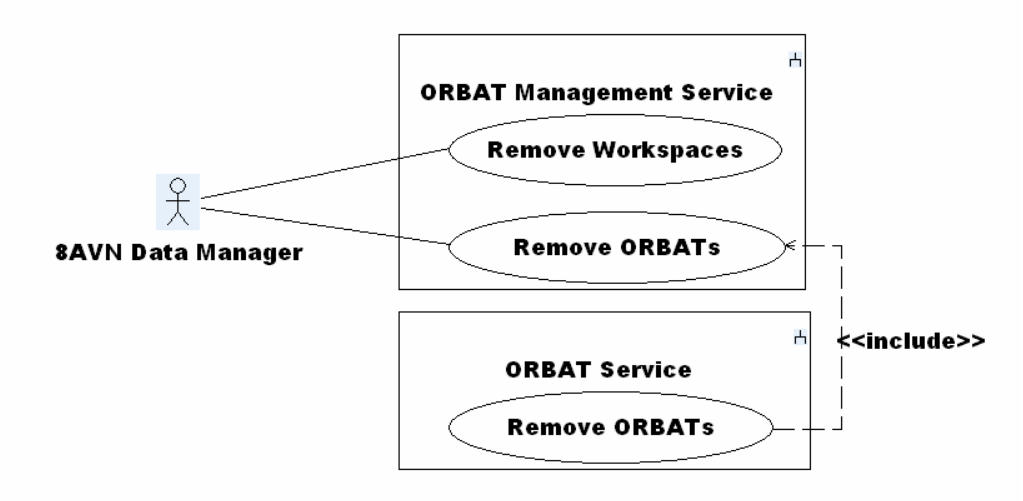


Figure 55 8AVN data manager's interaction with ORBAT tools for operation 2

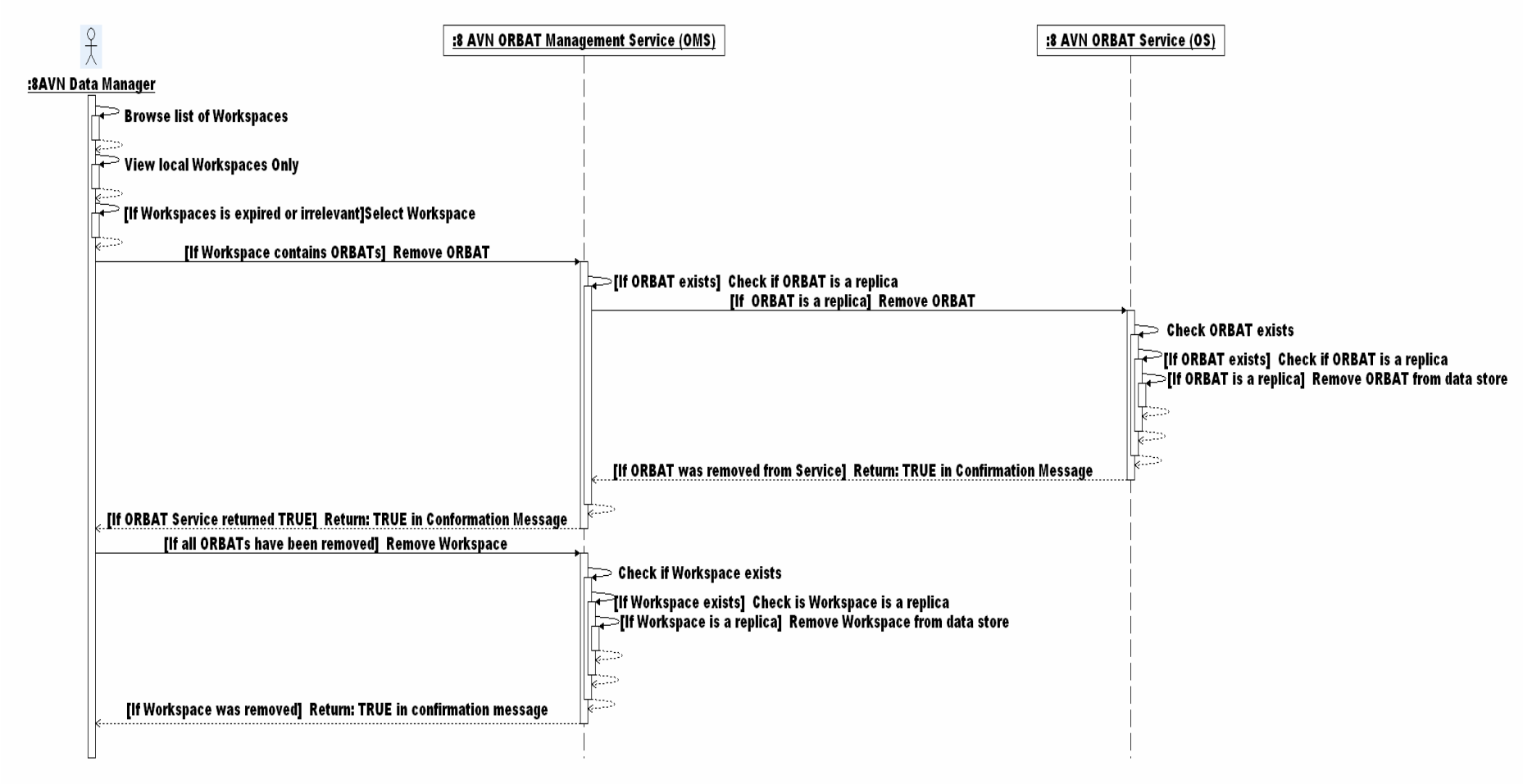


Figure 56 8ANV data manager's interaction with ORBAT tools

Operation 3: Move Authoritative ORBAT

The data manager proceeds to move the offending master Workspace and its managed ORBATs off the 8ANV deployed services to the 8AVN_Fixed ORBAT Service and JCSS_Fixed ORBAT Management Service. The SOPs dictate that original copies should reside here when not deployed.

> N.B. The Workspace is being moved to the JCSS_Fixed ORBAT Management Service, as that is the service the 8AVN_Fixed ORBAT Service is connected to. If 8AVN_Fixed were connected to the HQAST_Fixed Services, the Workspace would reside there instead.

The data manager would execute the move by using the OMS Client to copy the Workspace to the new ORBAT Management Service. Once this operation has been completed, the ORBATs that exist on the 8AVN_deployed ORBAT Service and are managed by this Workspace would be copied to the 8ANV_Fixed ORBAT Service. The target Workspace for the ORBATs being copied will be set to the Workspace that was copied to the JCSS_Fixed ORBAT Management Service.

Once the Workspace and ORBATs have been copied to the other services, the versions on the 8AVN deployed services are deleted; the data manager would use the OMS Client to complete this operation.

> N.B. When the Workspaces and ORBATs are copied they receive a new service ID and GUID. This information is stored in the fully qualified id element, which is attached to the Workspace/ORBAT. All links that other Workspaces have to these Workspaces and ORBATs will no longer be able to be resolved.

Figure 57 shows how the 8AVN data manager interacts with the ORBAT tool to move an authoritative ORBAT from one service to another. Figure 58 and Figure 59 show the interaction with the system in more detail.

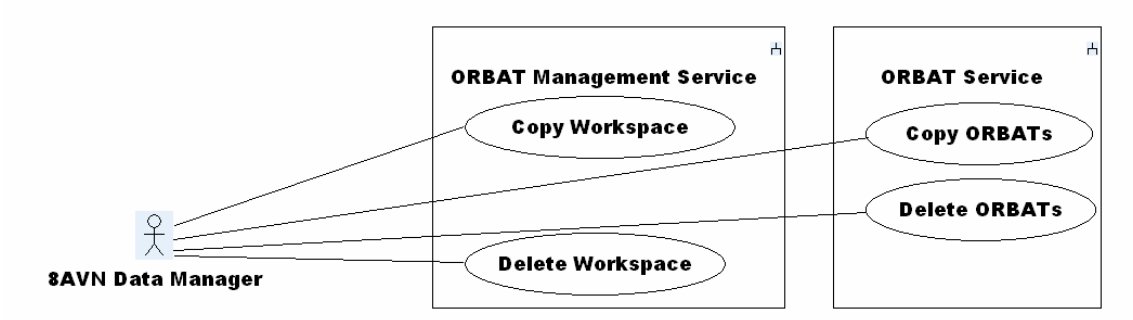


Figure 57 Use of ORBAT tools to move authoritative ORBAT

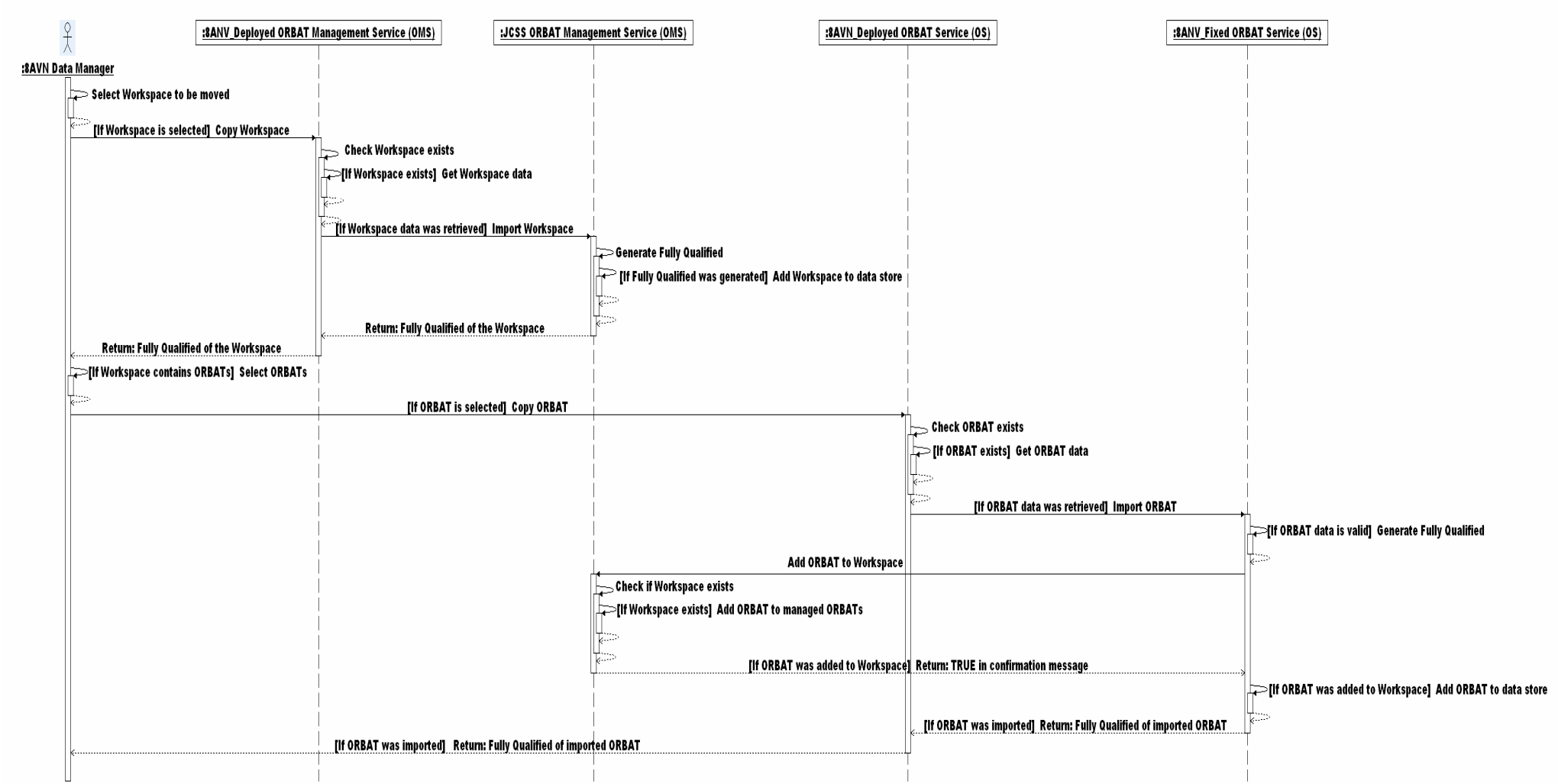


Figure 58 Interaction between data manager and services to move an authoritative ORBAT (part one)

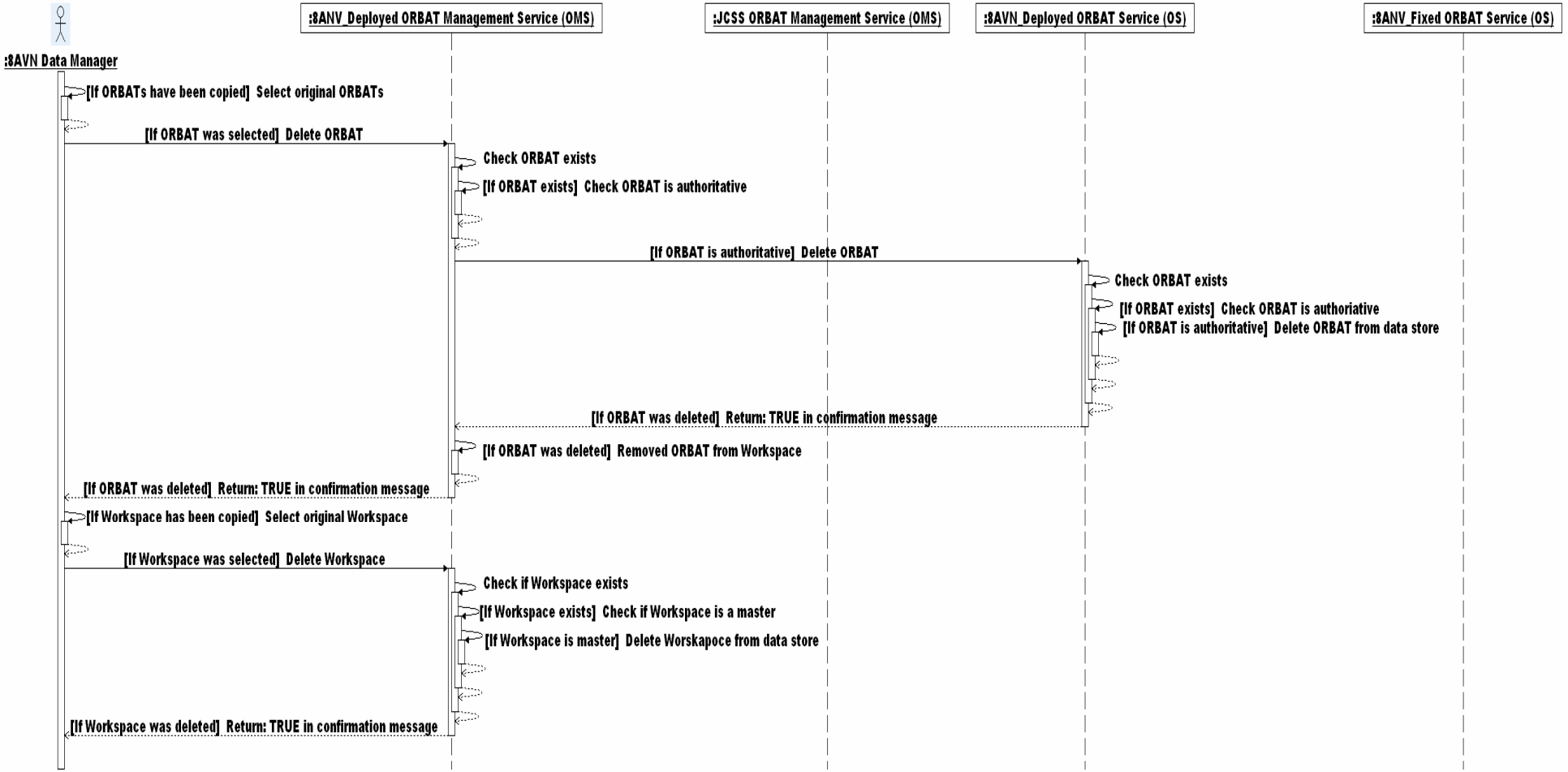


Figure 59 Interaction between data manager and services to move an authoritative ORBAT (part one)

Operation 4: Replicating Operational ORBATs

As part of the planning process the data manager has been notified that the Workspace containing the ORBATs for this operation is called MRO 333. The OMS Client is used to locate the Workspace and replicate it to the 8AVN_Deployed ORBAT Management Service. Table 23 contains the metadata for the MRO 333 Workspace.

Table 23: MRO 333 Workspace metadata

Workspace Name:	MRO 333
Description:	Military Response Options for Service Assisted Evacuation.
Derived From Template:	False
Is Template:	False
Master:	True
Intended Use:	Operational
Capability Constraint:	OLOC
ORBAT Hostility Allowed:	100001 (Assumed Friend) 100005 (Friend)

The data manager also finds and replicates the 8AVN_PLOC Workspace to the server to be deployed. Table 24 shows the metadata for the original 8ANV_PLOC Workspace that exists on the 8AVN_Fixed ORBAT Management.

Table 24 8AVN_PLOC Workspace metadata on 8AVN fixed OMS.

Workspace Name:	8ANV_PLOC
Description:	Present Level of capability for 8th Aviation Regiment.
Derived From Template:	TRUE
Derived From Template Id:	1234501787
Derived From Template name:	ADF AVIATION REGIMENT
Is Template:	False
Master:	True
Intended Use:	Operational
Capability Constraint:	OLOC
ORBAT Hostility Allowed:	100001 (Assumed Friend) 100005 (Friend)

Table 25 shows the metadata of the 8AVN_PLOC Workspace replica. Notice that the master attribute has been set to false.

Table 25 8AVN_PLOC Workspace metadata on 8AVN Deployed OMS.

Workspace Name:	8ANV_PLOC
Description:	Present Level of capability for 8th Aviation Regiment.
Derived From Template:	TRUE
Derived From Template Id:	1234501787
Derived From Template name:	ADF AVIATION REGIMENT

Is Template:	False
Master:	False
Intended Use:	Operational
Capability Constraint:	OLOC
ORBAT Hostility Allowed:	100001 (Assumed Friend) 100005 (Friend)

The original version of the MRO 333 Workspace held on the JCSS_Fixed OMS is the master version; the OMS Client is used to change it to FALSE and set the 8AVN_Deployed version to be the master. The 8AVN_PLOC Workspace on 8AVN_Deployed ORBAT Management Service also needs to be made the master, thus requiring the Workspace on 8AVN_fixed OMS to be changed to a replica or non-master version.

The ORBATs that were replicated onto the 8AVN_Deployed ORBAT Service also need to be made authoritative.

Figure 60 and Figure 61 show how the 8AVN data manager interacts with the ORBAT tools to replicate the operational ORBAT and make the appropriate Workspaces and ORBATs authoritative.

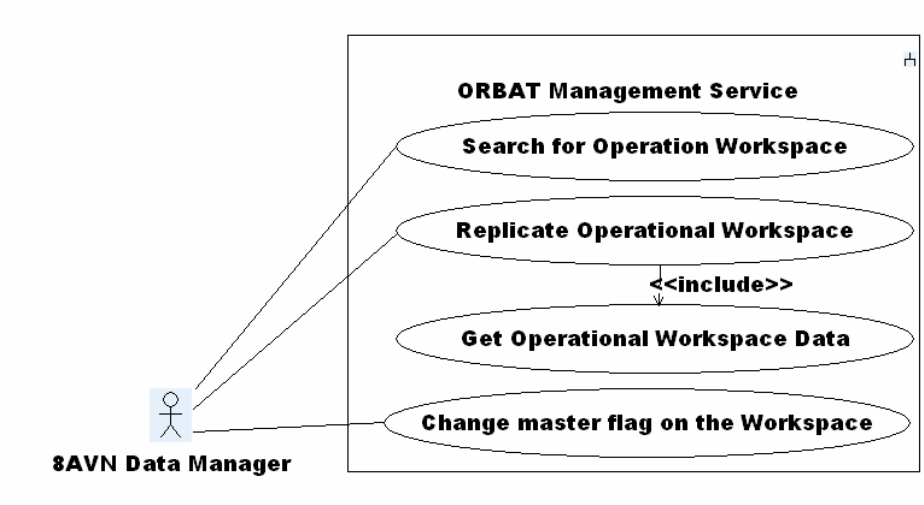


Figure 60 Using ORBAT tools to replicate operational Workspaces.

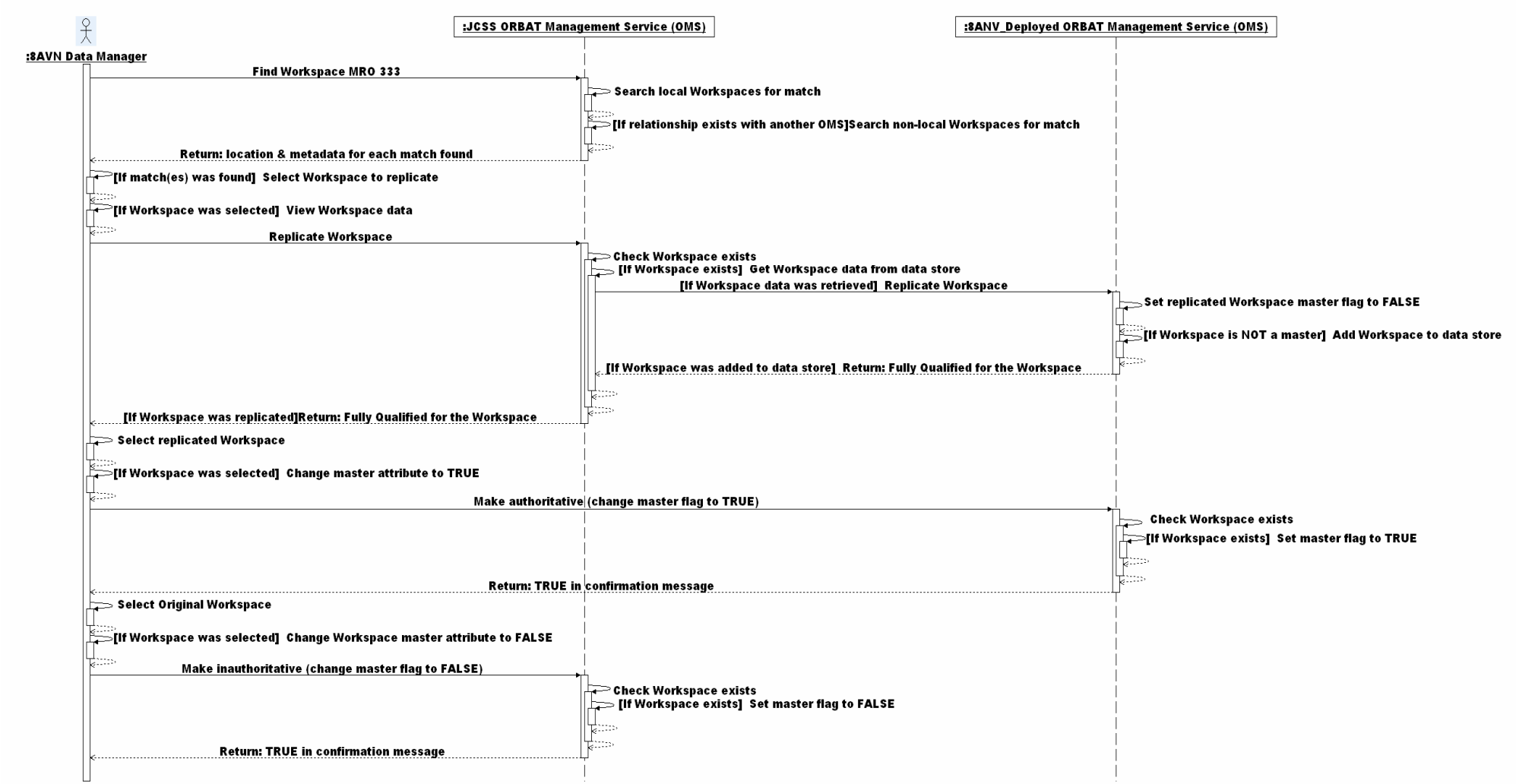


Figure 61 8AVN data manager invoking methods to replicate operational ORBATs

Operation 5: Set 8AVN_Deployed to Deployed.

The data manager is almost ready to deploy. He does not want the data on his newly loaded 8AVN_Deployed OMS to be accessible from the fixed network; the OMS Client will be used to set the status of the 8AVN_Deployed OMS (and connected ORBAT Services) to "Deployed – Not Available". The deployed attribute is set for the current date and time; the deployed to attribute is set for the anticipated end of the deployment in three months time.

The OMS Client is used to determine which ORBAT Services are connected to the 8AVN_Deployed OMS. The data manager then proceeds to set their status to "Deployed – Not Available" with the start and end dates corresponding to those on the OMS.

Figure 62 and Figure 63 shows the steps involved and the interaction between the 8AVN data manager and ORBAT tools in order to indicate that the services are deployed.

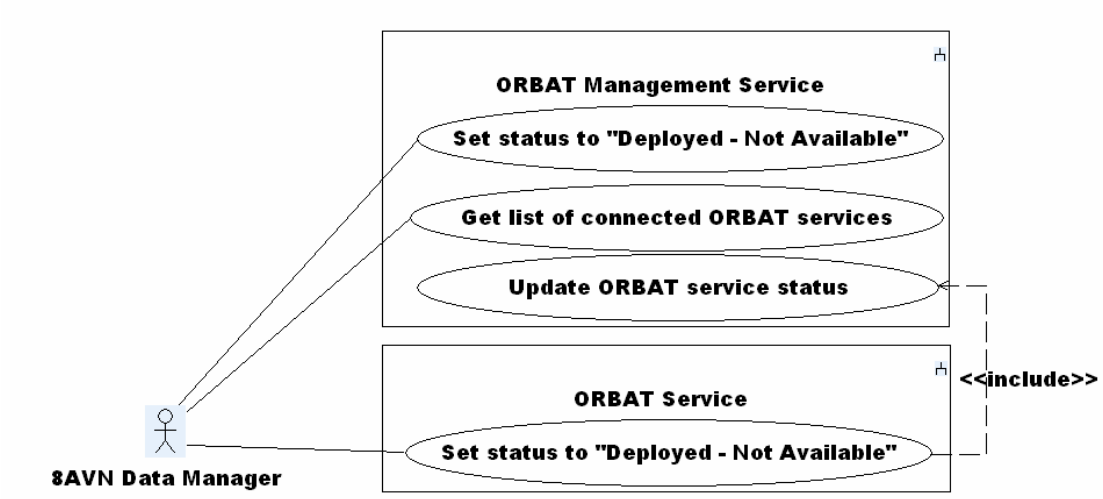


Figure 62 Deploying ORBAT tools for operation

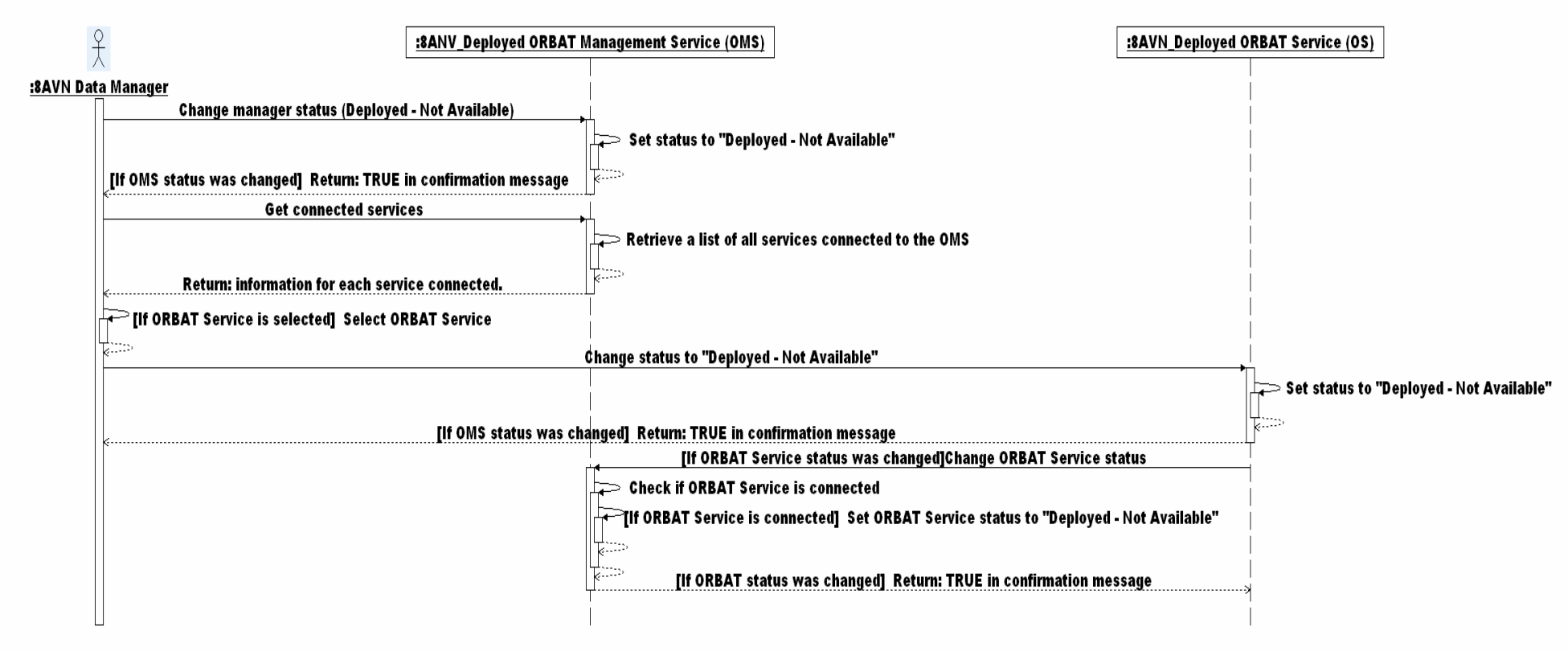


Figure 63 Steps for deploying a service.

Operation 6: Check Data Availability

The data manager has one final check to perform; the manager has been instructed that they may not gain access to the deployed JCSS network until a given period after their arrival. It will be important during this phase that they remain self-sufficient. The OMS Client is used to check that all the data for the operational Workspace (MRO 333 Workspace) and the 8AVN_PLOC Workspace is loaded onto the 8AVN_Deployed services.

To achieve this the OMS Client opens the nominated Workspaces and ORBATs. The data manager checks to see if a copy of the Workspaces and ORBATs is held locally on the 8AVN_Deployed ORBAT Service and OMS.

> N.B. All Workspaces and ORBATs that are referenced also need to exist on the local services and the references to the non-local copies need to be removed.

If the Workspace/ORBAT selected does not exist locally, the data manager selects the copy option in the client to have the additional information copied onto their server. The service(s) containing the 8AVN_Deployed ORBAT Server and the 8AVN_Deployed OMS may now be shut down and deployed.

Figure 64 shows the operations that need to be performed with the ORBAT tools in order to check that all data is available on the services that are being deployed.

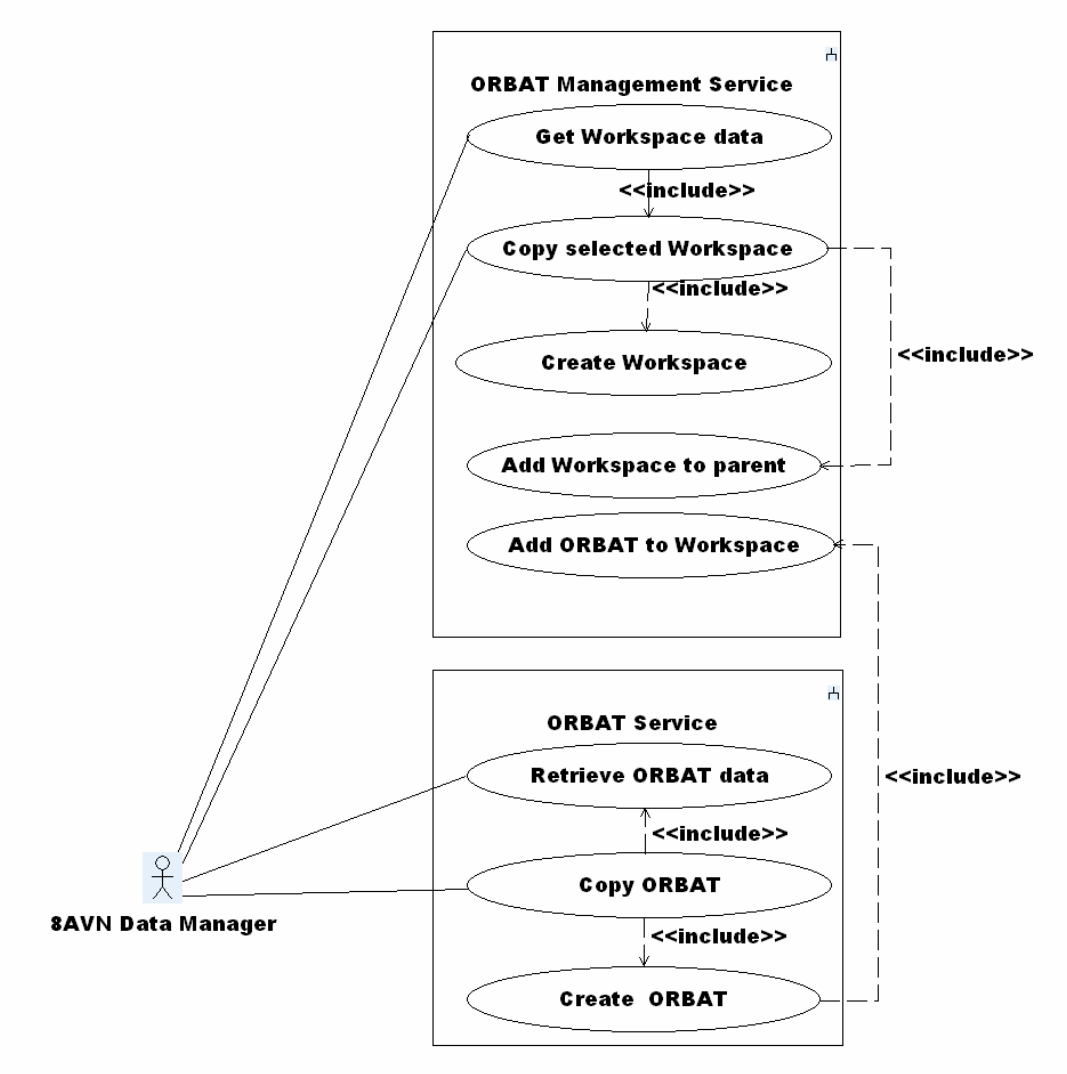


Figure 64 Using the ORBAT tools to check data availability

The following figures show the operations specified in the diagram above, in more detail.

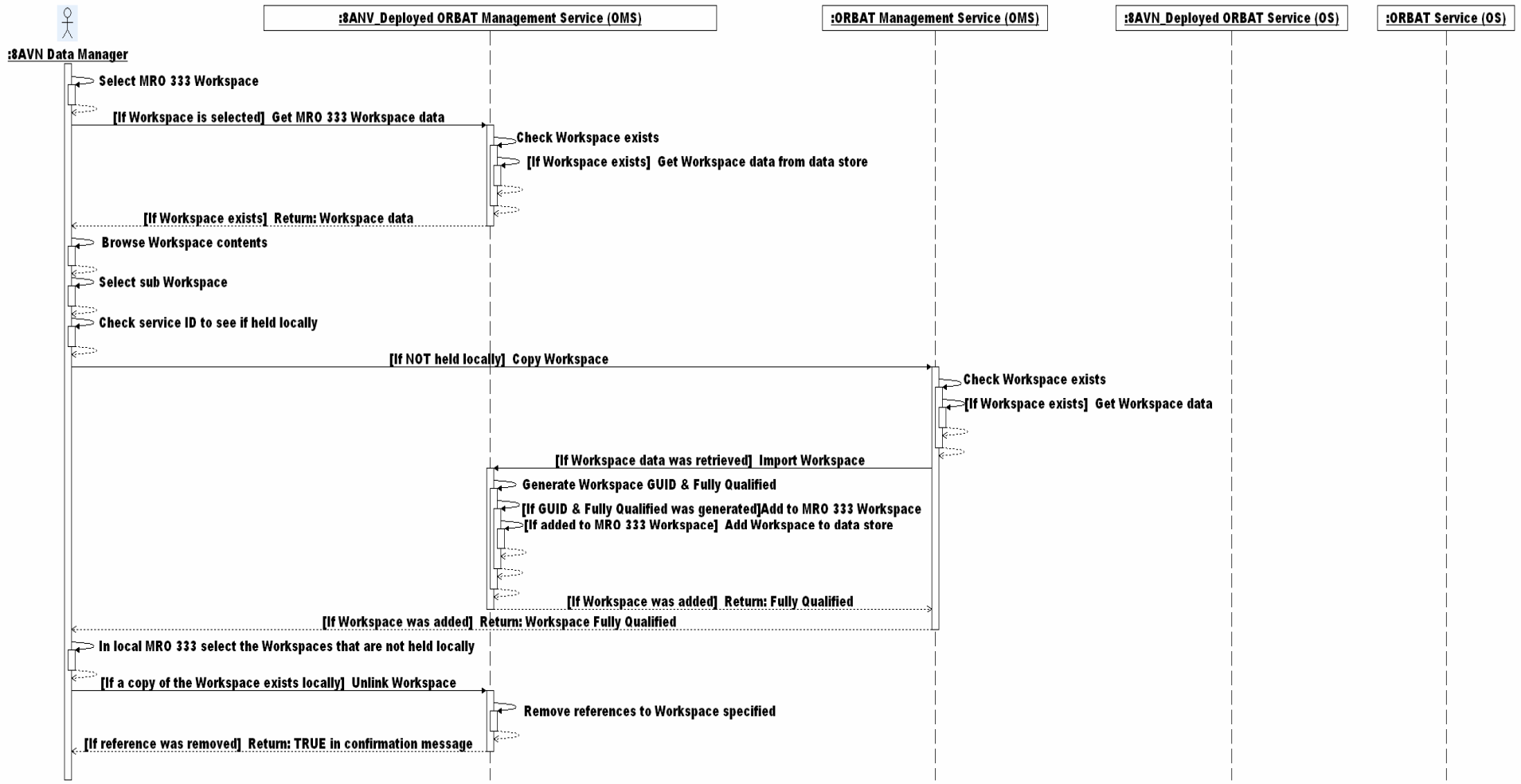


Figure 65 Invoking methods on the ORBAT services to check data availability (part 1)

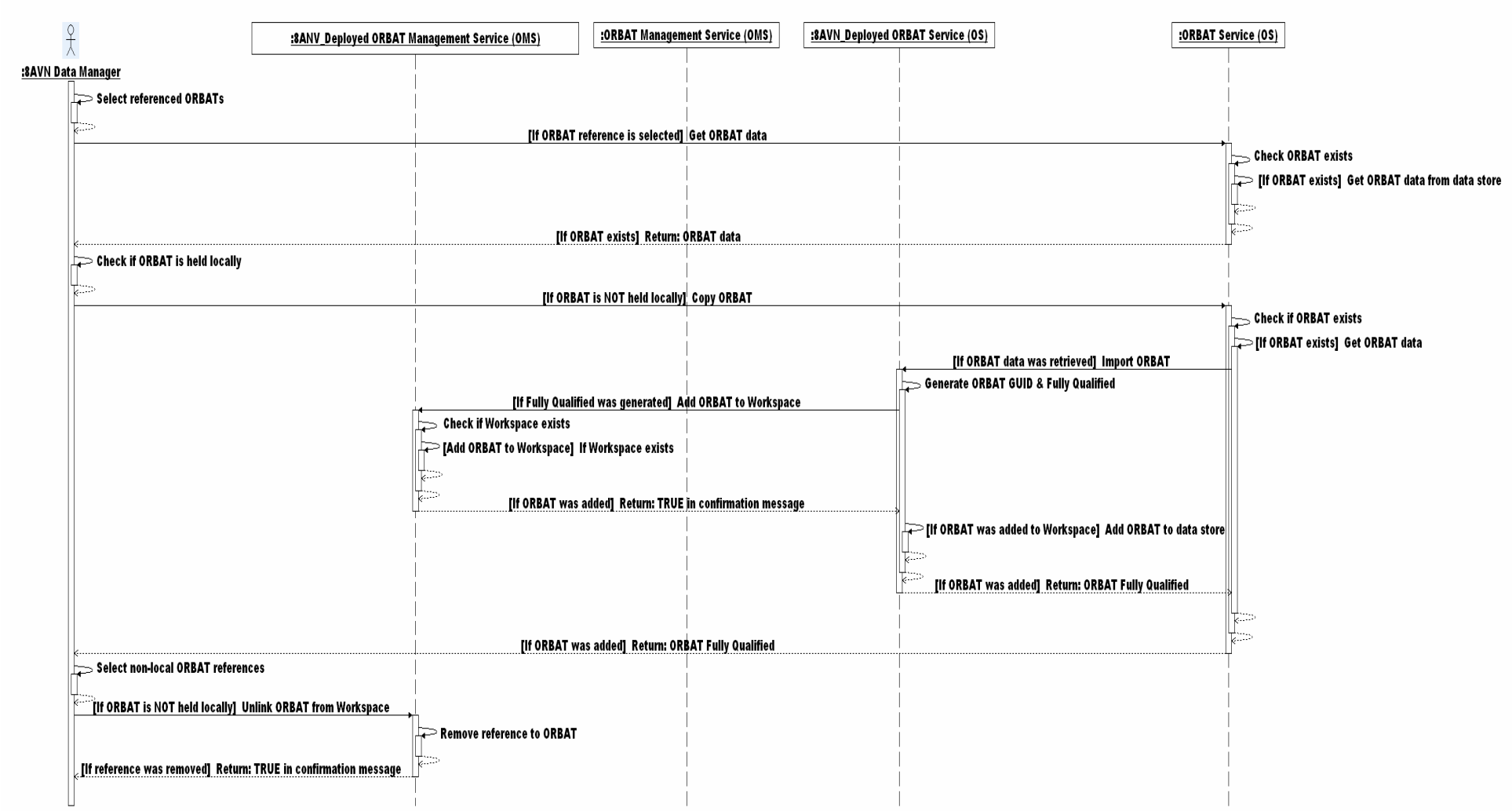


Figure 66 Invoking methods on the ORBAT services to check data availability (part 2)

Phase 2: Deployment

Operation 7: Connect the OMS

The 8AVN contingent has arrived in theatre; communications have been established with the DJFHQ via the JCSS network. Prior to establishing communications, the systems have been running stand-alone for several days.

The data manager is required to establish a connection with the DJFHQ_Deployed OMS to allow 8AVN staff to access other ORBAT information held by the task force. To achieve this they use the OMS Client to open the settings for the 8AVN_Deployed OMS and make the following changes.

Table 26 shows the relationships amongst the services that need to be removed. Table 27 shows the entries that need to be added.

Table 26 Services to be unregistered

Source OMS	*Target OMS*	*Indexes Replicated*
8AVN_Deployed	JCSS_Fixed	All local Workspaces
JCSS_Fixed	8AVN_Deployed	All local Workspaces

Table 27 Services to be registered

Source OMS	*Target OMS*	*Indexes Replicated*
8AVN_Deployed	DJFHQ_Deployed	All local Workspace
DJFHQ_Deployed	8AVN_Deployed	All local and non-local Workspaces

A connection with the DJFHQ server has now been established; the two servers now exchange indexing information, giving users at 8AVN visibility of the information available from the DJFHQ. In addition, because the DJFHQ OMS already receives updates from the JCSS fixed server and forwards them on, the 8AVN staff has visibility of Workspace and ORBAT metadata from the fixed network.

It is important to remember that at this point only the indexes are being replicated; if 8AVN users need access to an ORBAT or Workspace held on another service, the OMS Client will connect to that service and retrieve the data from there, exactly the same as in the fixed network.

Figure 67 shows the methods the 8AVN data manager would interact with in order to establish and remove relationships between services. Figure 68 and Figure 69 show the order and steps the 8AVN data manager should follow when setting up the relationships between ORBAT Management Services.

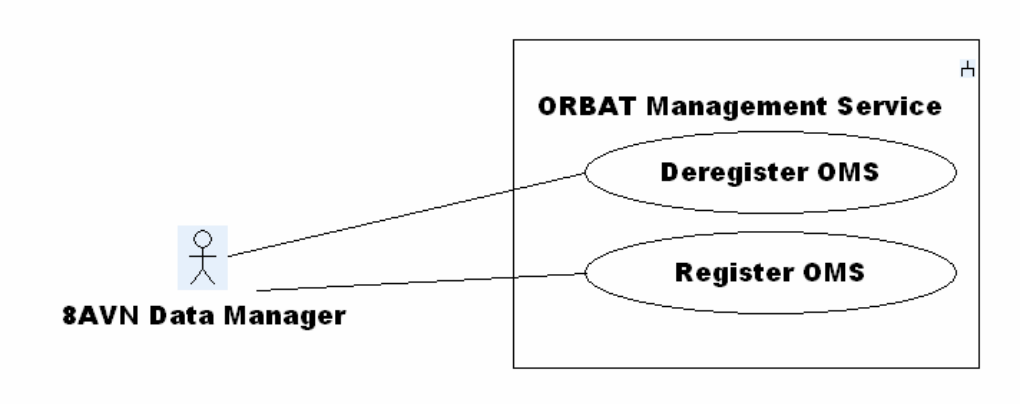


Figure 67 Procedure for creating relationships between ORBAT Management Services.

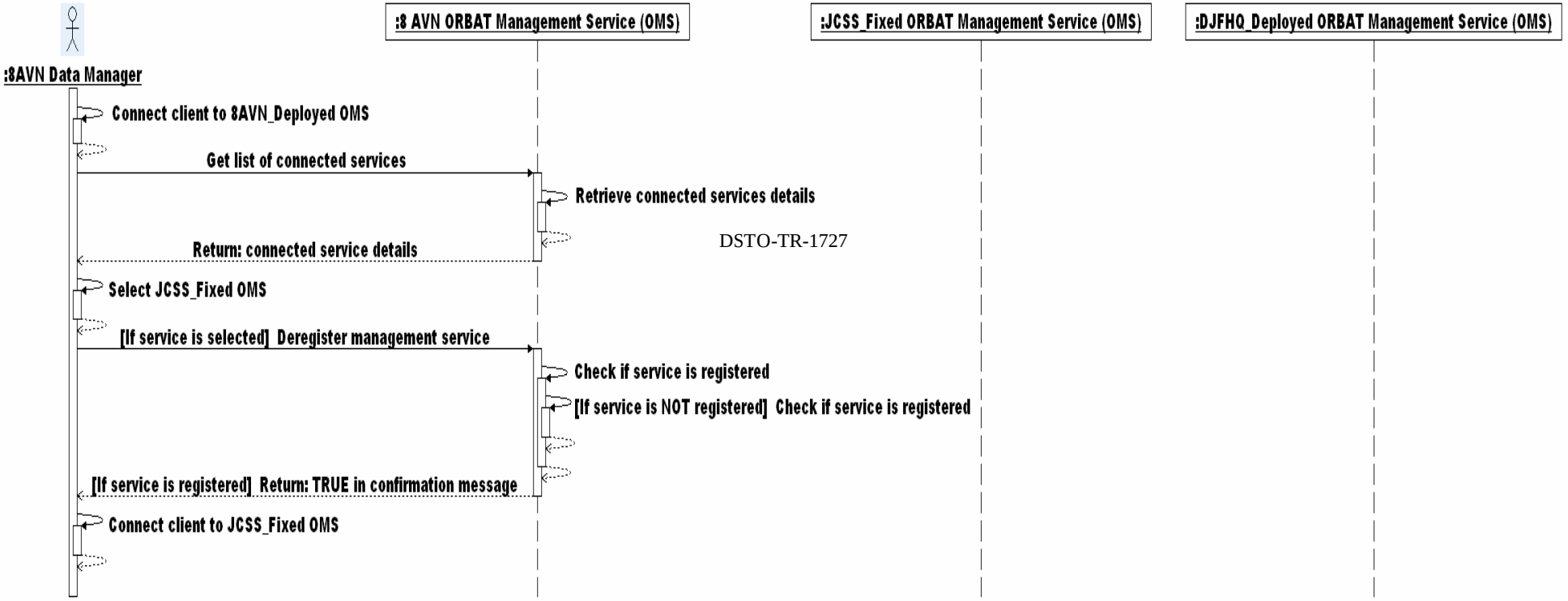


Figure 68 Invocation of methods on services to register/deregister services (part one)

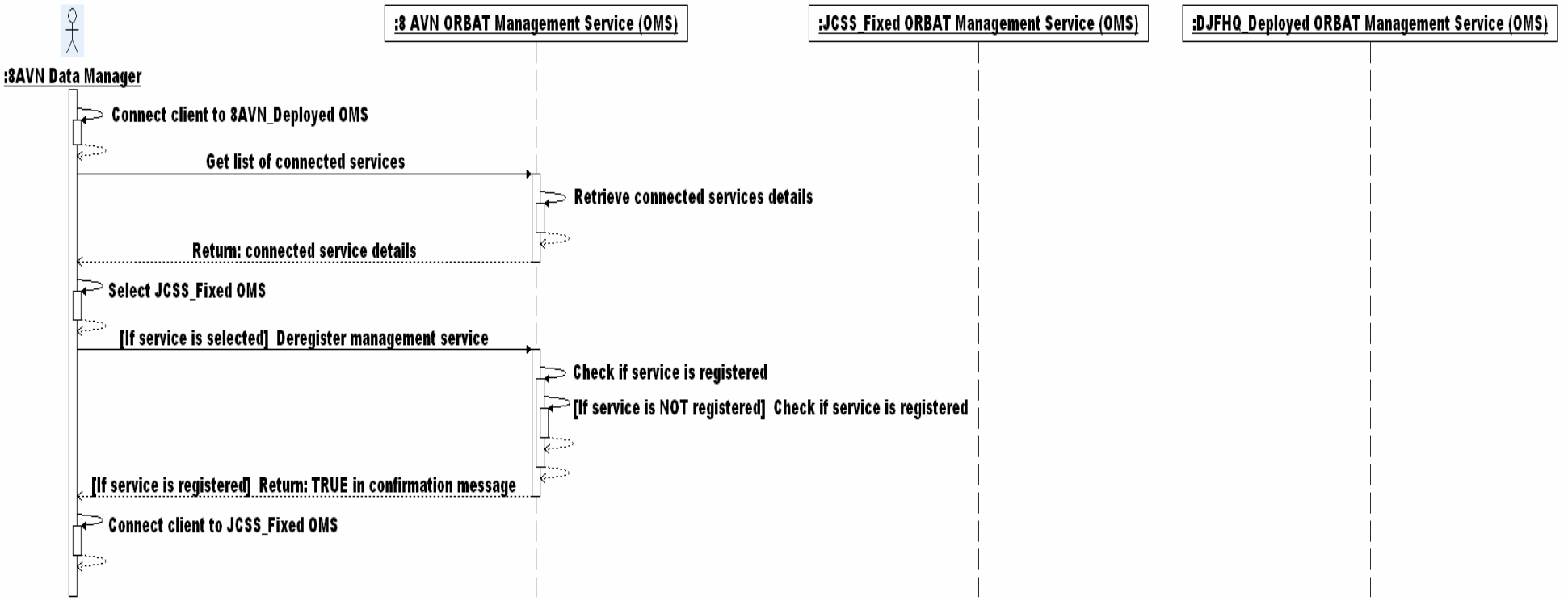


Figure 68 Invocation of methods on services to register/deregister services (part one)

Operation 8: Dealing with Multiple Authoritative Copies of ORBATs

As part of the replication of indexing information between OMSs, each service periodically checks to see if there are multiple authoritative copies of the same ORBAT or Workspace in the system. As a new relationship has been established, this process has been triggered on the 8AVN_Deployed OMS.

An OMS Client alert has informed the data manager that it has detected multiple authoritative copies of an ORBAT that is stored on one of its ORBAT Services. The data manager is presented with the names of the offending ORBATs and may choose to set the local copy to unauthoritative or to contact the data manager of the other offending services.

There are three ORBATs on the 8AVN_Deployed Services that have a conflict with ORBATs on other services. The conflicting ORBATs are the RHQ ORBAT, HQ A SQN ORBAT and HQ B SQN ORBAT. The metadata for each of the ORBATs is shown in the series of tables below.

Table 28 conflicting RHQ ORBAT metadata

ORBAT Name:	RHQ ORBAT
Exercise:	False
Derived From Template:	False
Is Template:	False
Intended Use:	Operational
Authoritative:	True
Fictional:	False
Capability Constraint:	OLOC
Unit hostility allowed:	1000001 (Assumed Friend) 1000005 (Friend)

Table 29 conflicting HQ A SQN ORBAT metadata

ORBAT Name:	HQ A SQN ORBAT
Exercise:	False
Derived From Template:	False
Is Template:	False
Intended Use:	Operational
Authoritative:	True
Fictional:	False
Capability Constraint:	OLOC
Unit hostility allowed:	1000001 (Assumed Friend) 1000005 (Friend)

Table 30 conflicting HQ B SQN ORBAT metadata

ORBAT Name:	HQ B SQN
Exercise:	False
Derived From Template:	False
Is Template:	False
Intended Use:	Operational
Authoritative:	True
Fictional:	False
Capability Constraint:	OLOC
Unit hostility allowed:	1000001 (Assumed Friend) 1000005 (Friend)

The data manager decides that the RHQ ORBAT in the BLUE FORCES (MRO 333 child Workspace) Workspace should not be authoritative. The OMS Client is used to change the authoritative attribute on the ORBAT to FALSE. The manager of the remaining two ORBATs is contacted to resolve the other conflicts.

The metadata shown in Table 31 reflects the changes the 8AVN data manager has made to resolve the conflict. Note that the Authoritative attribute is now set to false.

Table 31 modified RHQ metadata for RHQ ORBAT

ORBAT Name:	RHQ ORBAT
Exercise:	False
Derived From Template:	False
Is Template:	False
Intended Use:	Operational
Authoritative:	False
Fictional:	False
Capability Constraint:	OLOC
Unit hostility allowed:	1000001 (Assumed Friend) 1000005 (Friend)

Figure 70 shows the methods the 8AVN data manager would use to resolve conflicts between data. Figure 71 shows the steps the 8AVN data manager would take in more detail.

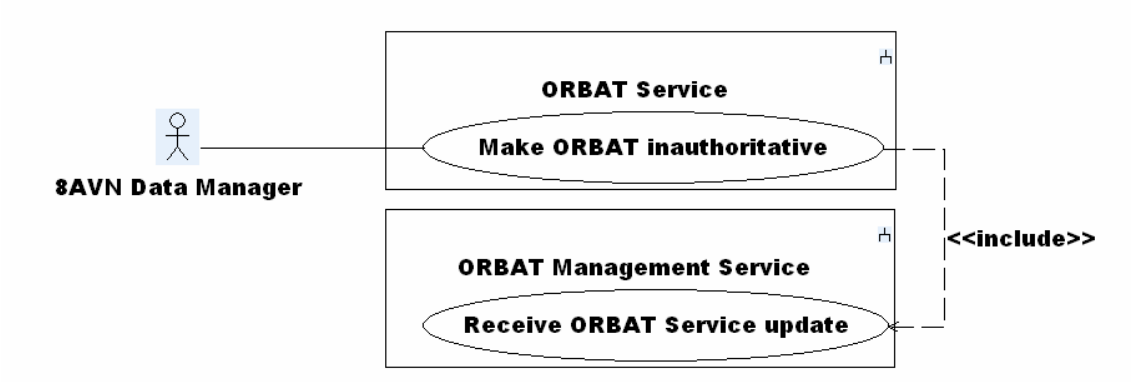


Figure 70 8AVN data manager's interaction with ORBAT tools to resolve conflicts.

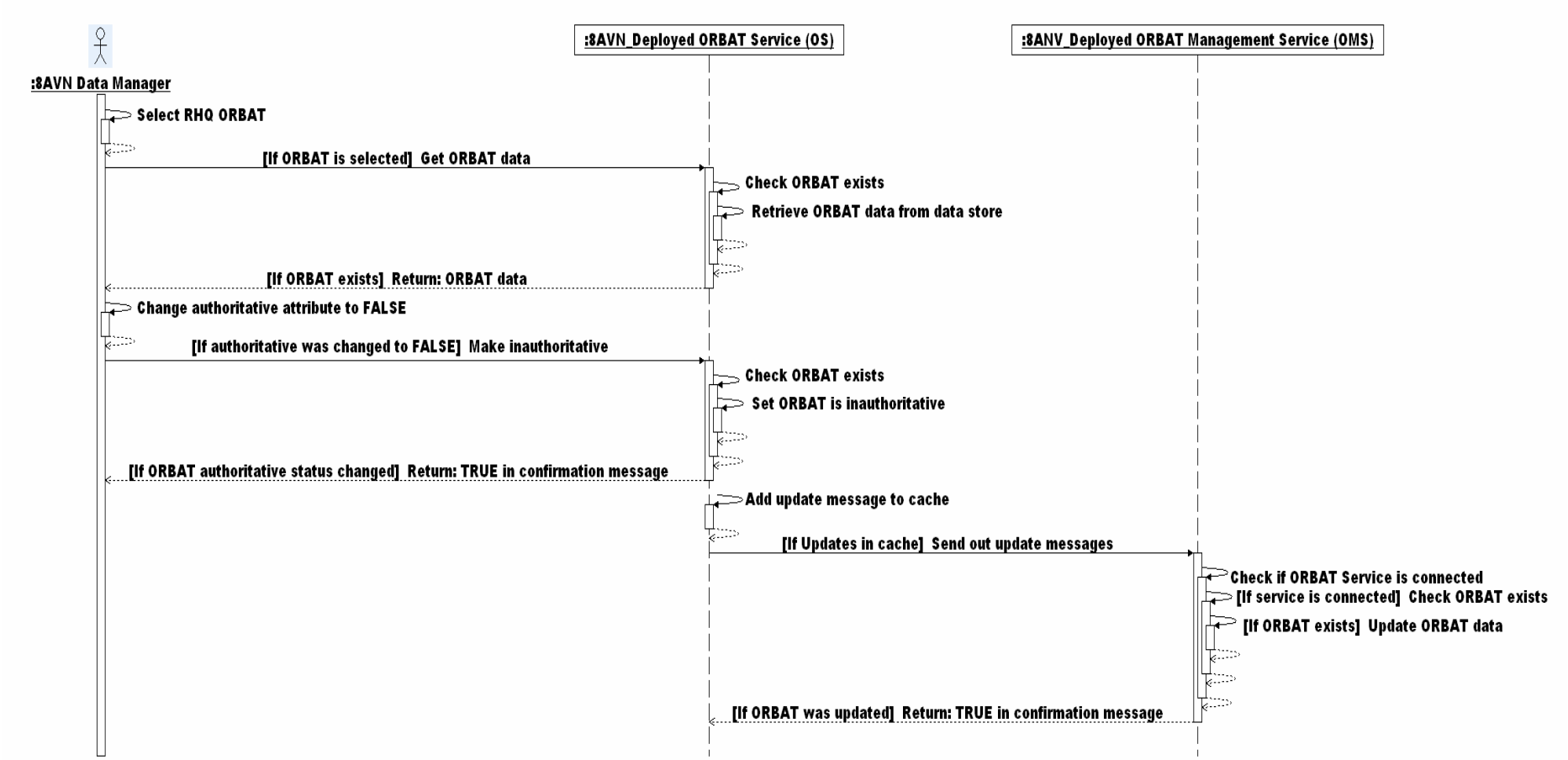


Figure 71 8AVN data manager's use of ORBAT tools to resolve conflicts in data

8.4 Planning for ADF Exercises.

This use case describes how the ORBAT tools can be used to assist ADFWC personnel in preparing for an exercise.

Preconditions:

Existence of a planning Workspace for the exercise (called EXERCISE ELEPHANT 03) on the JCSS Fixed ORBAT Management Service.

Table 32 shows the metadata of the EXERCISE ELEPHANT 03 Workspace.

Table 32 EXERCISE ELEPHANT 03 Workspace metadata

Workspace Name:	EXERCISE ELEPHANT 03
Description:	ADF Exercise focusing on peace keeping and humanitarian relief.
Derived From Template:	False
Is Template:	False
Master:	True
Intended Use:	Plan
Capability Constraint:	OLOC
Effective From:	14/07/03
Expires On:	31/7/03
ORBAT Hostility Allowed:	N.B. No constraints, indicating that the Workspace can hold Workspaces/ ORBATs of any hostility.

The EXERCISE ELEPHANT 03 Workspace should contain two child Workspaces called BLUE FORCE and RED FORCE. As indicated by the name, the BLUE FORCE Workspace will hold the data for the friendlies used within the exercise. Table 33 shows the metadata for the BLUE FORCE Workspace.

Table 33 BLUE FORCE Workspace metadata

Workspace Name:	BLUE FORCE
Description:	Contains Blue force or data for friendlies used in exercise ELEPHANT 03.
Derived From Template:	False
Is Template:	False
Master:	True
Intended Use:	Plan
Capability Constraint:	OLOC
Effective From:	14/07/03
Expires On:	31/7/03
ORBAT Hostility Allowed:	1000001 (Assumed Friend) 1000005 (Friend)

The RED FORCE Workspace will hold data for the enemy forces. The metadata for this Workspace is shown in Table 34 below.

Table 34 RED FORCE Workspace metadata

Workspace Name:	RED FORCE
Description:	Contains RED FORCE or enemy data for Exercise ELEPHANT 03.
Derived From Template:	False
Is Template:	False
Master:	True
Intended Use:	Plan
Capability Constraint:	OLOC
Effective From:	14/07/03
Expires On:	31/7/03
ORBAT Hostility Allowed:	1000002 (Assumed Hostile) 1000006 (Hostile)

N.B. The creation of Workspaces and child Workspaces has already been shown in an earlier use case. Please see the Land Headquarters Force Assembly Use Case for further details.

The ADFWC ORBAT Service has established a relationship with the JCSS Fixed ORBAT Management Service (See Figure 72 below). The personnel at ADFWC will now have visibility of current operations Workspaces so they can check that units required for the exercises are not deployed.

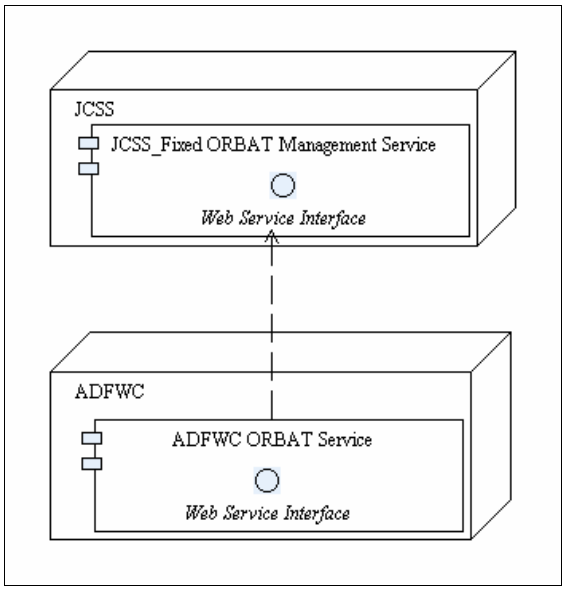


Figure 72 ADFWC ORBAT Service deployment

Operations to Occur:

1. Create an EX ELEPHANT BLUE FORCE ORBAT within BLUE FORCES Workspace.
2. Search for current PLOC information for units required for the exercise.

3. Copy selected PLOC unit data from PLOC ORBAT into BLUE FORCE exercise ORBAT.
4. Populate RED FORCE ORBAT for purpose of EXERCISE ELEPHANT.
5. Search operational Workspaces to see if/when units required are deployed.

Operation 1: Create EX ELEPHANT BLUE FORCE ORBAT in BLUE FORCE Workspace

The ADFWC Planner will use the ORBAT Client to create an ORBAT within the BLUE FORCES Workspace that will hold the unit data and structure for the BLUE FORCE team. The metadata for the EX ELEPHANT BLUE FORCE ORBAT is shown in the table below.

Table 35 EX ELEPHANT BLUE FORCE ORBAT metadata

ORBAT Name:	EX ELEPHANT BLUE FORCE
Exercise:	True
Derived From Template:	False
Is Template:	False
Intended Use:	Plan
Authoritative:	True
Fictional:	True
Capability Constraint:	OLOC
Unit hostility allowed:	1000001 (Assumed Friend) 1000005 (Friend)

Figure 75 shows the process that the ADFWC planner must go through to create the exercise ORBAT. Figure 74 shows this process in more detail.

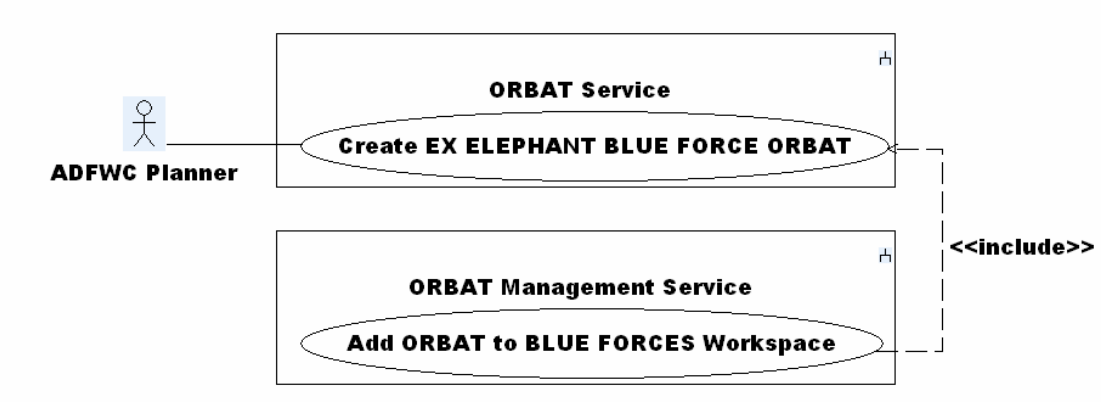


Figure 73 ADFWC planner's interaction with ORBAT tools for operation

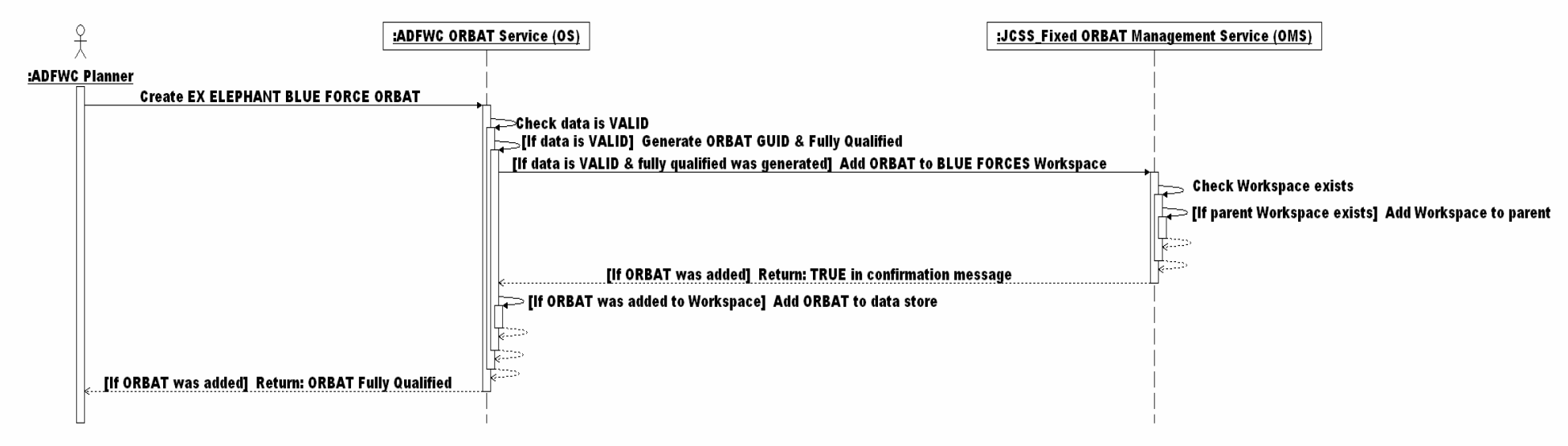


Figure 74 ADFWC Planner's interaction with ORBAT tools to create exercise ORBAT

Operation 2: Search for current PLOC information for units required for the exercise.

The ADFWC planner, in conjunction with other planners, has decided that some of the units in 2RAR (2nd Battalion Royal Australian Regiment) will be used to play as the BLUE FORCE. The specific sections of 2RAR that are to be utilised for the exercise are the four rifle companies, a support company (including the reconnaissance and communications specialists) and the battalion headquarters (including the Admin Company and Intelligence Section). Figure 75 below represents the intended structure for the EX ELEPHANT BLUE FORCE ORBAT.

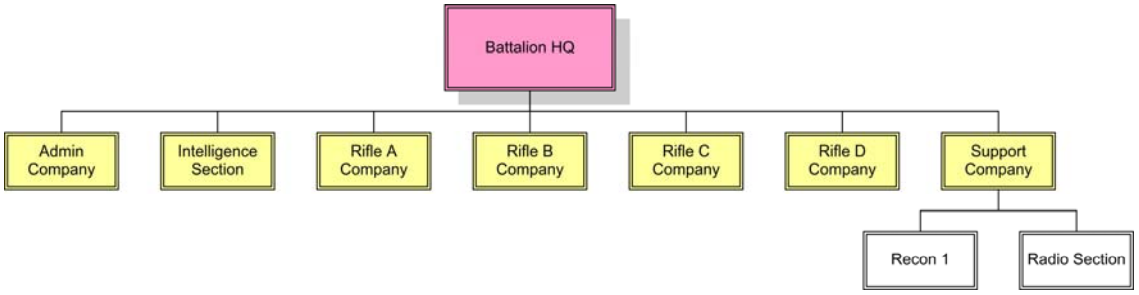


Figure 75 Intended EX ELEPHANT BLUE FORCE ORBAT structure

The ADFWC planner would use their ORBAT Service Client to search for the current 2RAR PLOC ORBAT.

> N.B. As all of the units being used in the exercises are from the same Battalion it is easier to locate their current PLOC data. If the planned exercise was utilising units from a variety of battalions the planner would need to search all ORBATs for a specific units data.

Figure 76 shows the ADFWC planner interacting with the ORBAT Management Service to locate the 2RAR PLOC ORBAT. Figure 77 shows this interaction in more detail.

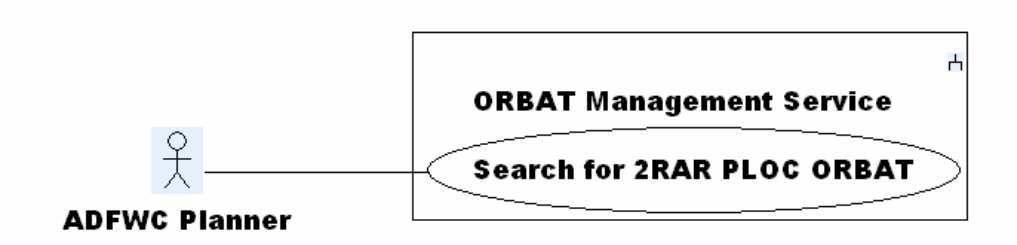


Figure 76 ADFWC planner interacting with ORBAT Management Service.

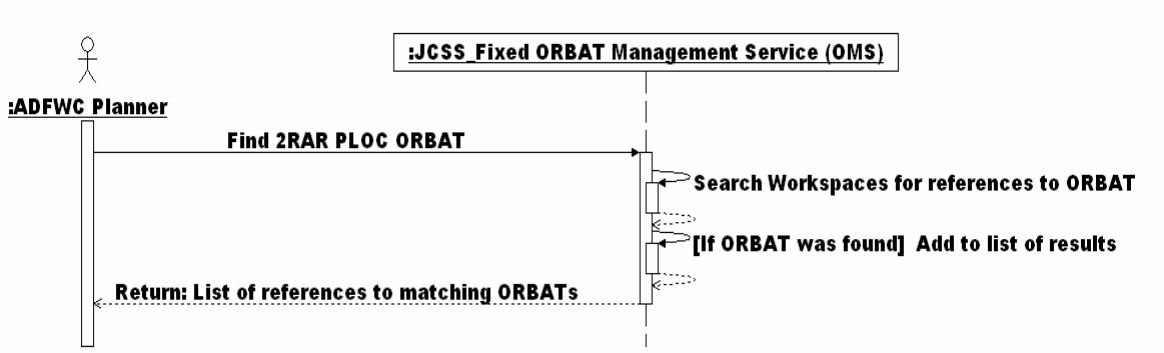


Figure 77 ADFWC planner searching for PLOC Workspace on OMS

Operation 3: *Copy selected ploc unit data from PLOC ORBAT into BLUE FORCE exercise ORBAT.*

After the PLOC ORBAT has been retrieved, the planner may proceed to use the client to copy the unit data for each of the companies and sections required for the exercise into the EX ELEPHANT BLUE FORCE ORBAT.

> N.B. When unit data is copied, a portion the hierarchy can be copied and inserted into the new ORBAT with an identical structure. Alternatively, each individual unit can be copied across and a new structure can be made. When individual units are copied, a parent unit must be specified unless they are a root unit. For example, Battalion HQ is a parent unit.

For the purposes of the exercise, Recon 1 and the Radio Section are not under their normal chain of command. Therefore, all units except those two may be copied together, making the structure identical to that of the 2RAR PLOC ORBAT. After these units have been copied across, the planner can continue to use the client to copy the remaining two units individually, specifying Support Company as the parent unit.

Figure 78 shows the methods the ADFWC planner would invoke in order to copy the unit data into the BLUE FORCE ORBAT.

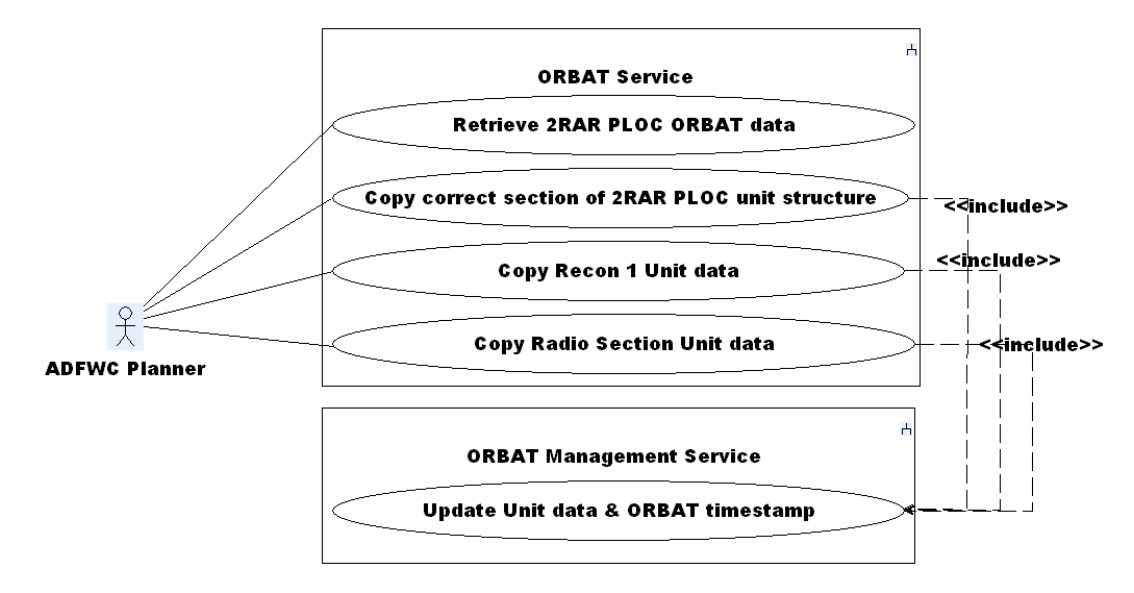


Figure 78 ADFWC planner interacting with ORBAT tools to copy unit data.

The following series of diagrams (Figure 79, Figure 80 and Figure 81) shows the ADFWC planner copying the unit data between the ORBATs in more detail. Notice that when an update is made to an ORBAT, an update message is added to the cache and later sent out to the OMS.

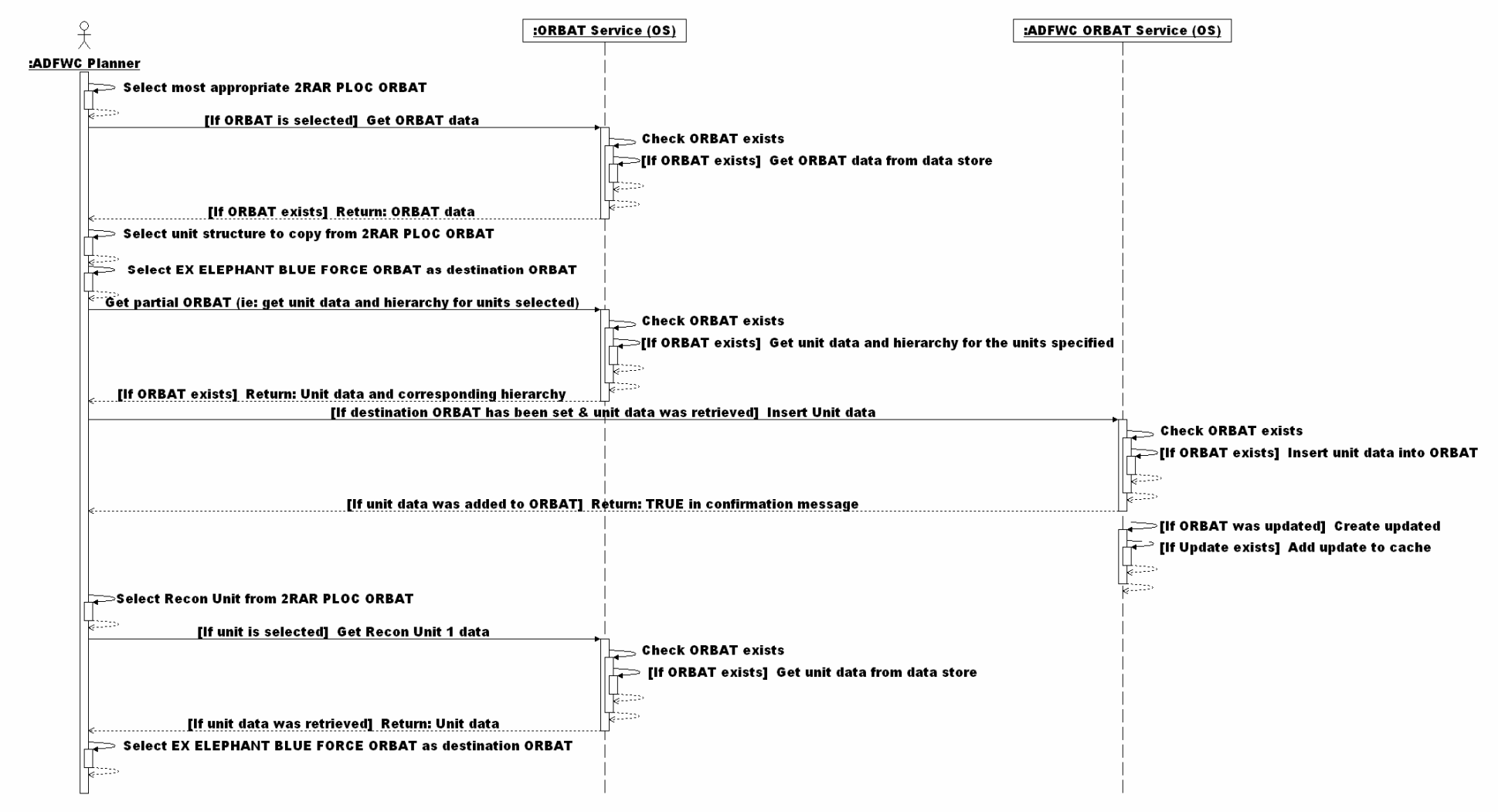


Figure 79 Copying unit data from one ORBAT into another (part 1)

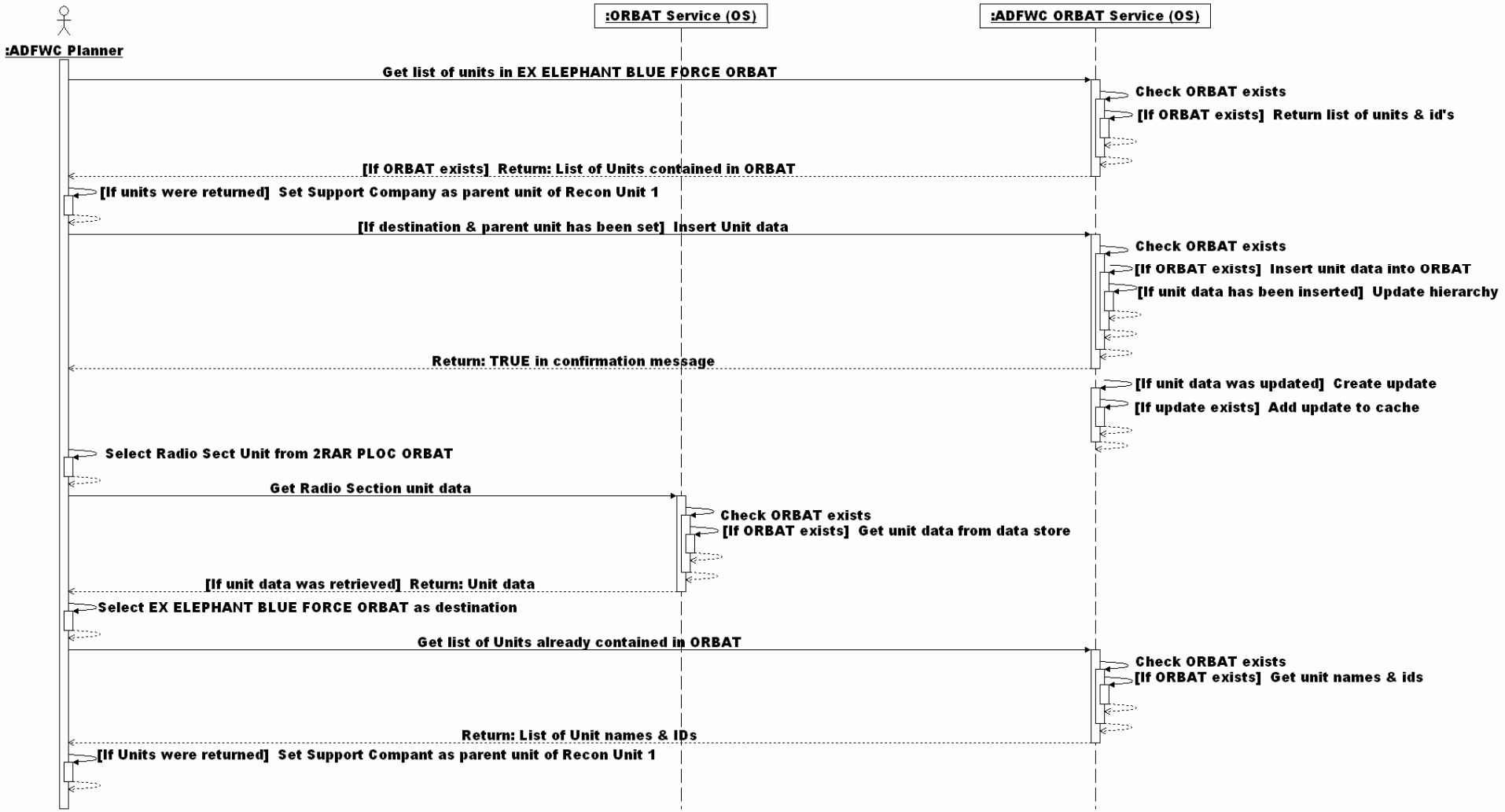


Figure 80 Copying unit data from one ORBAT into another (part 2)

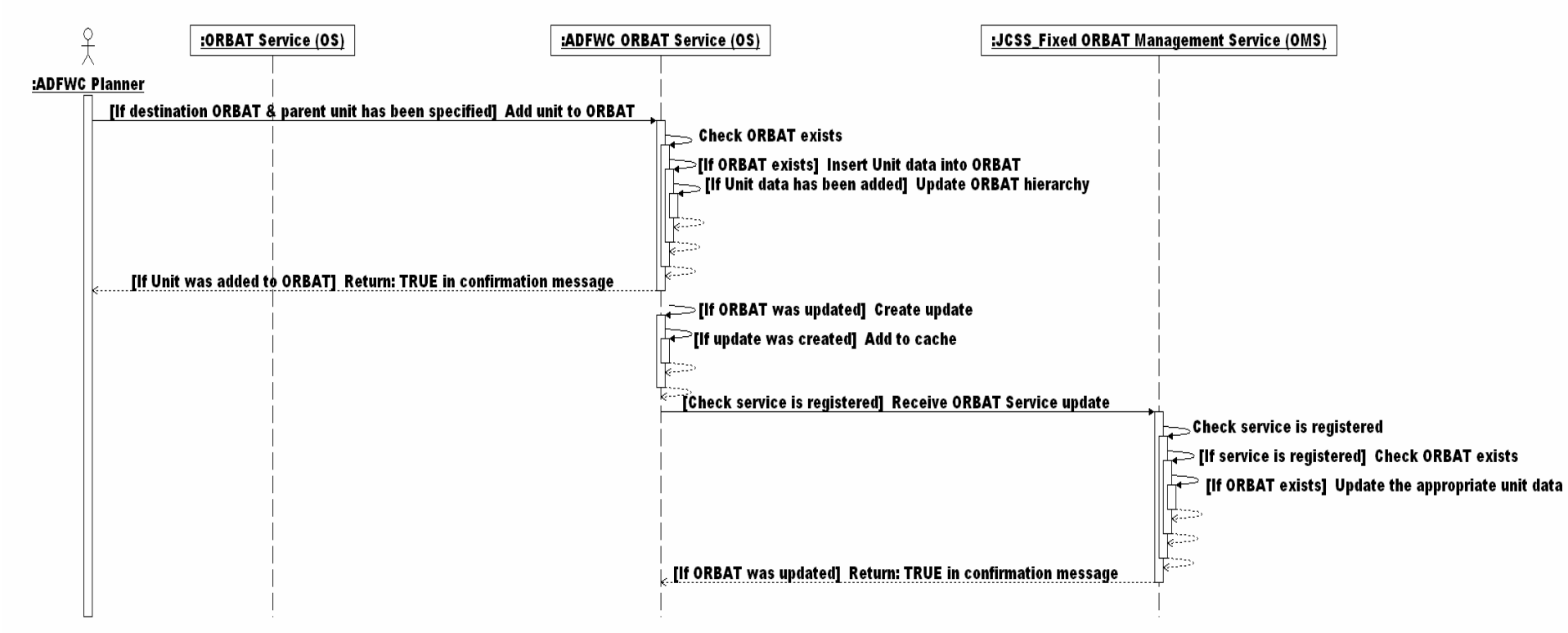


Figure 81 Copying unit data from one ORBAT into another (part 3)

Operation 4: Populate RED FORCE ORBAT for purpose of EXERCISE ELEPHANT.

An ORBAT is required to hold the data for the RED FORCE used during the exercise. The RED FORCE ORBAT will contain the hostile forces. It has been decided that 3RAR will be used as the RED FORCE. The ADFWC planner will search for the 3RAR PLOC ORBAT, retrieve the data and use the client to copy the ORBAT into the RED FORCES Workspace. Table 36 shows the 3RAR PLOC ORBAT metadata.

Table 36 3RAR PLOC ORBAT metadata

ORBAT Name:	3RAR PLOC
Exercise:	False
Derived From Template:	False
Is Template:	False
Intended Use:	Doctrine
Authoritative:	True
Fictional:	False
Capability Constraint:	PLOC
Unit hostility allowed:	1000001 (Assumed Friend) 1000005 (Friend)

The copy of the ORBAT will be renamed to EX ELEPHANT RED FORCE and other metadata on the ORBAT will be altered to reflect the new context it is being used in. Table 37 shows the modified metadata for the RED FORCE ORBAT (previously a 3RAR PLOC ORBAT copy).

Table 37 Ex ELEPHANT RED FORCE ORBAT

ORBAT Name:	EX ELEPHANT RED FORCE
Exercise:	True
Derived From Template:	False
Is Template:	False
Intended Use:	Plan
Authoritative:	True
Fictional:	True
Capability Constraint:	PLOC
Unit hostility allowed:	1000001 (Assumed Friend) 1000005 (Friend)

Figure 82 shows the ADFWC planner using the ORBAT tool to populate the RED FORCE ORBAT for the exercise. Figure 83 shows how the ORBAT tools are used in more detail.

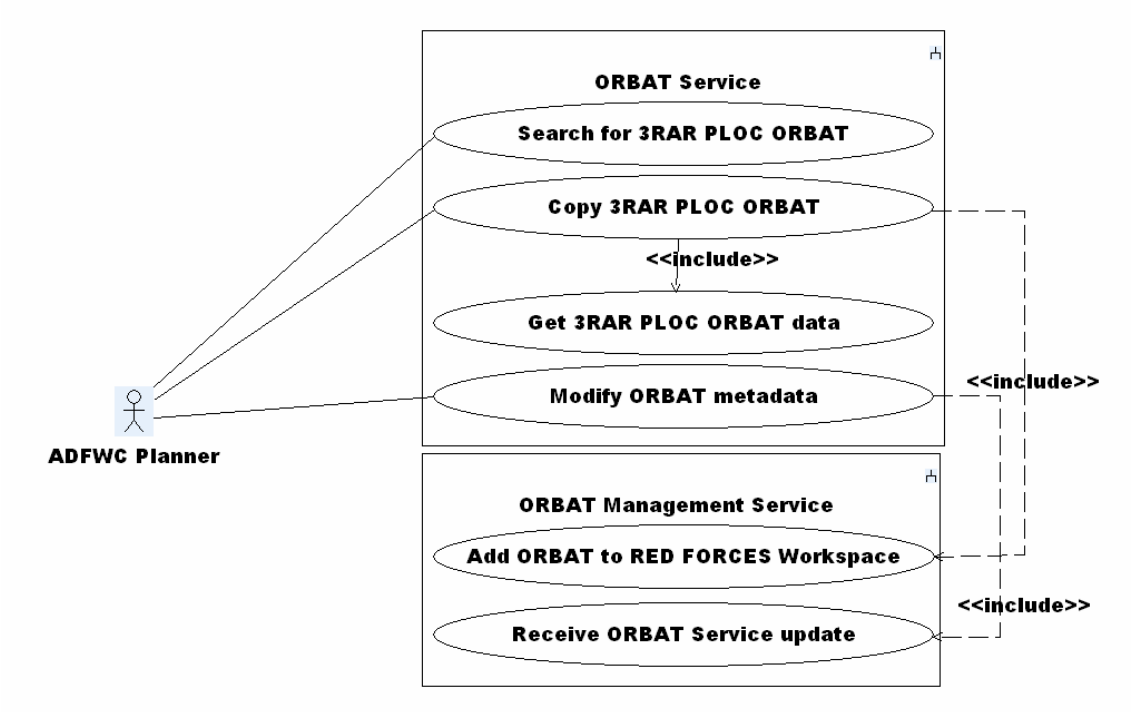


Figure 82 ADFWC planner populating RED FORCE

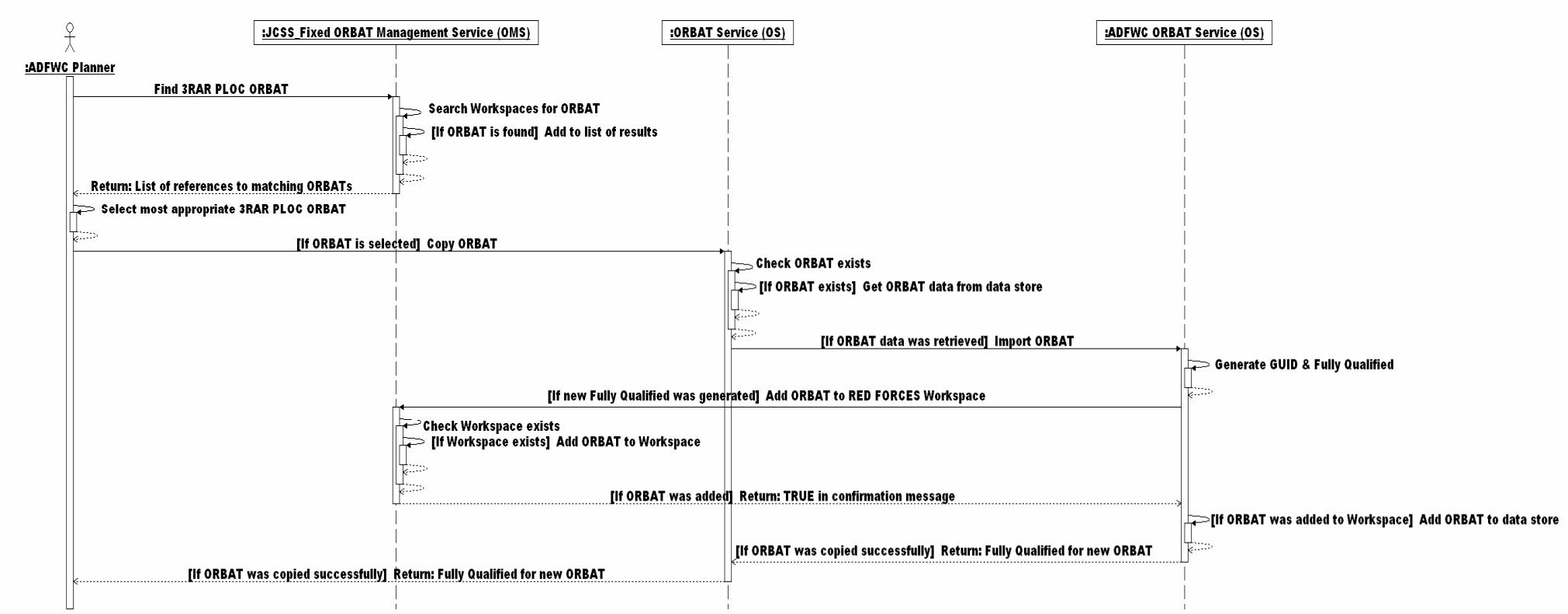


Figure 83 ADFWC planner copying ORBATs & populating exercise RED FORCE ORBAT

Operation 5: Search operational Workspaces to see if units required are deployed

Now that the forces for the exercise have been created, the planner performs one last check on the data. The operational Workspaces and ORBATs that are running at the same time as the exercise are searched for the units being used within the exercise, to ensure that the units in the exercise are available during that period.

N.B. Searching for units within ORBATs is a resource intensive search and should be used wisely.

The planner uses the client to search all ORBATs for the units involved in the exercise. A list of ORBATs these units are contained in will be returned.

N.B. Filtering the results of the search to exclude ORBATs not valid within the time of the exercise is a client specific function. The ORBAT Services do not provide this functionality.

Several ORBATs are returned that contain units being used within the exercise, however none of them conflict with the period the exercise is running.

Figure 84 shows the ADFWC planner using the ORBAT Service to search for units utilised in the exercise. Figure 85 shows the search process in more detail.

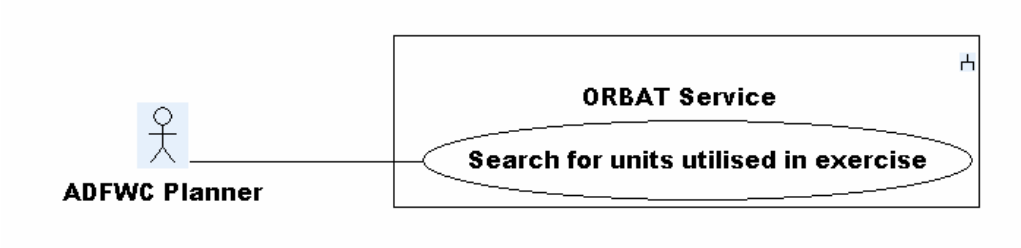


Figure 84 ADFWC planner searching ORBAT Services for units

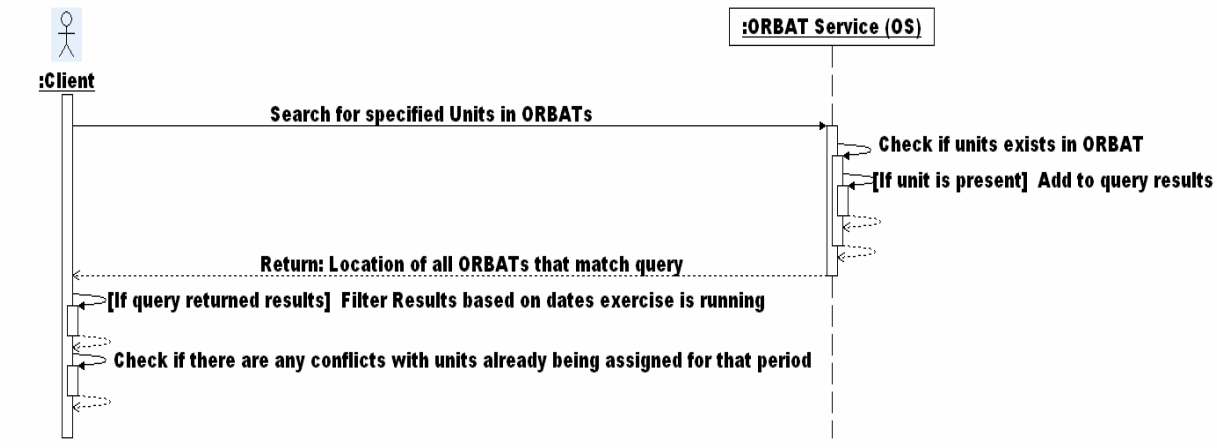


Figure 85 Searching ORBAT Services for units

9. Conclusion

The ORBAT Services Working Group has been an example of a successful collaboration between Defence and Industry. We have successfully used current generation industry standards to implement a complicated service that is at the core of C3I functionality. This effort should not be seen as producing a stand-alone product, for that was not it's intent. Rather this should be seen as the first successful step on the road to a Service Based Architecture for C3I Systems.

DSTO have been extolling the virtues of Service Based Architectures for many years. With the development and widespread acceptance of XML and the Web Services Family of standards, we have seen a significant shift from competition between middleware flavours such as COM and CORBA to near universal support for Web Services. This has for the first time nearly eliminated the risk of investing in a dead end technology. This view is now widely supported within the IT industry; the giants of the industry such as Microsoft, ORACLE and SAP continue to make massive investments in Web Services and Service-Orientated Architectures.

The current state of play is summarised neatly in Metagroups recent analysis [1] 'Bottom Line: Organizations that begin the transition to service-oriented architectures now will realize the benefits of those architectures immediately. If standards-based technologies are used to implement the architecture, those benefits will come with little risk to the organisation.'

Since the completion of this work the Web Services Interoperability (WS-I) Organisation has been formed to promote Web Service interoperability across platforms, applications and programming languages. It is comprised of many different software vendors, Web Services customers and anyone else with an interest in the area. WS-I have developed a series of resources to aid in the development of Web Services and provide a means to verify that services developed meet industry standards and WS-I guidelines.

The Basic Profile provides guidelines for the use of core Web Services specifications including WSDL, SOAP and UDDI. The ORBAT services should be modified to ensure they meet these guidelines as it will improve their interoperability with future services and simplify integration.

The WS-I Usage Scenario profile presents a series of Web Service design patterns that should be used and built upon when developing Web Services. The scenarios provide details about the different forms of messaging that can be implemented in a Web Service. Currently ORBAT service communication follows the Synchronous Request/Response pattern. An analysis should be performed to determine if this is the most appropriate form of communication for all functions offered by the system or if a different technique from the profile should be used. The usage scenario provides the developer with all the information they need to correctly implement the type of messaging desired.

In addition to the benefit of having highly interoperable services, development time can be accelerated through the use of the resources provided by WS-I. Furthermore, developers have clear and concise guidelines, which remove any ambiguity from the Web Services standard. Due to the changing nature of the Defence environment reduced development

time will help to ensure that requirements are being met as systems can be modified or implemented to meet additional requirements. Moreover, it also means that the cost of developing such a service is reduced.

10. Recommendations

In this section DSTO provides recommendations for the adoption of XML, XML Schema, XSLT and Web Services Document Style Encoding.

10.1 Standardise on XML and Web Services as the default and preferred integration technology within the JCSE

We recommend that XML and Web Services be adopted as the core integration technology within in the JCSE. XML should become the normal mechanism for exchanging information between JCSE applications.

10.2 Investigate the development of a Military Messaging to XML Bridge

Considerable time and expense has been invested in existing Military Messaging formats and infrastructure. These formats share much in common with XML. It is technically possible to bridge the gap between Military Messaging and XML by extending an existing message parser. This may offer a quick low cost mechanism to make a broad range of military information accessible and manageable within COTS applications. We recommend that this opportunity be investigated for its utility and integration benefits.

We do not propose that current applications of Military Messaging be replaced with XML. Rather we wish to expose this information as XML to a broader range of applications within the JCSE. Message formats will continue to be an important information exchange mechanism between subsystems of the JCSE.

10.3 Establish a searchable XML Schema and XSLT Repository

Defence should prepare itself for the deployment and widespread use of XML technologies by focusing on the development of common XML Schemas and an appropriate range of XSLT transformations.

Building an XML Schema and XSLT repository will be a significant task, potentially there are going to be literally thousands of schemas and transforms. Keeping track of this XML is going to be a challenge in itsself. Therefore it is recommended that CSS Branch in consultation with the Office of the CIO identify the appropriate metadata to be stored within schemas and XSLT to assist in their management.

The format for this metadata should be published as a documentation standard and included as an acceptance criteria in JCSE contracts that use XML (whether developed or acquired as part of COTS software). This standardised documentation will form the link to the Defence Common Language repository.

This metadata will be used within the repository for search and identification purposes, it will not affect the actual contents of the Schemas and Transformations. Taking this step will ease the information management burden associated with an enterprise level XML repository.

10.4 Establish an XML Data Management Focus Group

The role of this group will be to develop workable procedures to manage the transition to XML. A key responsibility of this group would be to review XML usage in the JCSE and ensure that ongoing processes are in place to identify common data schema.

The goal here is to move towards standardisation by ensuring that common data is identified and used as part of a larger corporate dictionary. In the case of the ORBAT services we have included additional data tags (name/value pairs), to allow end users to exchange data elements we haven't yet thought of in our XML Schema. This approach gives us a great degree of flexibility now with the opportunity to gain interoperability via data standardisation in a minimalist and ongoing approach.

As a first effort the group could look at using the NATO MIP as an excellent starting point. There has been considerable effort put into the ontology of the MIP. We have shown with the ORBAT Services that it is possible to reuse elements of this model as XML. It should be noted that at the time of writing the MIP team are not as advanced in their usage of XML as this effort. It may be better to move ahead with our own efforts instead of waiting for further developments in this space. The use of XSLT will ensure that even if we do make our own start we will still be able to achieve interoperability.

DSTO has also been involved with the Coalition Theatre Logistics Task. This task has also adopted the XML data exchange approach. Considerable corporate knowledge has been developed in that effort. It is appropriate to consider extending the scope of that task to Australian C2 Systems. The staffs involved on this task are well placed to make a strong contribution to the development of the JCSE.

10.5 Prepare to use the XML features of Office 2003

XML is now the ubiquitous information processing and exchange mechanism. With the release of Microsoft Office 2003 the power of XML is now fully realisable on the Desktop. Whilst we can expect that there will be some delay in introducing these technologies into the COE we should be using this time to plan for the integration of these products. This approach would help speed the consistent adoption of XML within the C2 System.

An end user search interface should be developed for Microsoft Office to allow access to the enterprise XML repository. This would allow an end user to search the repository for the XML representations. This component would be used by all office applications to allow a user to search or navigate the common language repository for the appropriate XML to use in a given situation. Standard XSLT transforms associated with particular schemas would also be available from the search tool.

Additional Office components could be built to automatically fill out common XML, for example a planner should be able to label all documents created during the planning for

an operation with the name of that operation. An Office Plugin that allowed the planner to set the operation and security classification and monitor all documents created and used would be a useful way to make information management easy to perform and relevant to the end user.

10.6 Participate in the deployment planning for UDDI Services

Web Services natively use UDDI as their directory service. Third party products that have web service interfaces will be acquired sooner or later and will need a UDDI Infrastructure. UDDI seems to fall within the management remit of Secret Services; we expect it to be rolled out along with Microsoft 2003 Server.

CSS Branch must take an active interest in the development of a distributed UDDI Infrastructure. When considering the deployment of UDDI there must be consistent procedures and policies regarding the registering, identification and maintenance of services. This will impact on JCSE development and integration issues. Special consideration will need to given to federating distributed UDDI Services. Policy to support deployed environments must be included as a requirement of the initial plan. This policy should focus on streamlining the rapid deployment and redeployment of services in support of operations. Failure to address this issue in a coherent way will have an adverse impact on the command support systems.

10.7 Establish a CSS Branch/Industry Application Security Working Group

Within the CSS systems there is currently no discernable Security Architecture. Application level security arrangements are ad hoc and aimed at satisfying accreditation requirements on an individual basis. There is no coherent vision for securing C2 applications and data. Little thought has been given to the consequences of this approach. Individual security arrangements lead to a proliferation of security enforcement techniques and a burgeoning systems administration load. The normal day-to-day business of C2 involves the application of operational security, particularly in the area of closely held planning. Moving to an integrated CSS implies that we will be opening up some elements of the existing CSS, thus our reliance on traditional systems boundaries as security barriers will no longer be appropriate.

The role of the Application Security Working Group would be to foster and develop a unified C2 view of Applications security. The group would allow open dialog on security techniques and standards.

11. Acknowledgements

DSTO would like to take this opportunity to thank our Defence industry participants for the effort and insight that they have provided in this process. In particular ADI, Aspect, Tactica Data Systems and Microsoft have all made significant contributions to this work; in many cases participants have put in their own unpaid effort to ensure the success of this

endeavour. We would also like to thank Tenix for their early contribution during the difficult stage of problem definition and conceptualisation of the problem space.

---

1 D. Sholler, "Practical Approaches to Service-Oriented Architecture – Meeting the Demand Today and Tomorrow", META Group, 208 harbour Drive, Stamford, CT 06902, White Paper, November 2003.

DISTRIBUTION LIST

ORBAT Services Design Version 1.0

Glen Coomber, Carly Frobes and Philip Hawthorne

AUSTRALIA

DEFENCE ORGANISATION

Task Sponsor

Director General Command and Support Systems Branch, DMO	1

S&T Program

Chief Defence Scientist FAS Science Policy AS Science Corporate Management Director General Science Policy Development	shared copy
Counsellor Defence Science, London	Doc Data Sheet
Counsellor Defence Science, Washington	Doc Data Sheet
Scientific Adviser to MRDC, Thailand	Doc Data Sheet
Scientific Adviser Joint	1
Navy Scientific Adviser	Doc Data Sheet & Dist List
Scientific Adviser - Army	Doc Data Sheet & Dist List
Air Force Scientific Adviser	Doc Data Sheet & Dist List
Scientific Adviser to the DMO	Doc Data Sheet & Dist List

Information Sciences Laboratory

Chief Command & Control Division	Doc Data Sheet
Research Leader Command Decision Environments Branch	1
Research Leader Information Enterprises Branch	1
Research Leader Joint Command Analysis Branch	Doc Data Sheet
Research Leader Intelligence Information Branch	Doc Data Sheet
Head Human Systems Integration	Doc Data Sheet
Head Information Exploitation	Doc Data Sheet
Head Effects-Based Modelling and Analysis	Doc Data Sheet
Head Information Systems	1
Head Distributed Enterprises	Doc Data Sheet
Head Joint Operations Analysis and Support	Doc Data Sheet
Head Command Concepts and Architectures	Doc Data Sheet
Head Command Process Integration and Analysis	Doc Data Sheet
Head Intelligence Analysis	Doc Data Sheet
Glen Coomer (Task Manager/Author)	1
Carly Forbes (Author)	1

Philip Hawthorne (Author)	1
Publications and Publicity Officer, C2D/EOC2D	1 shared copy

DSTO Library and Archives

Library Edinburgh	1
Defence Archives	1
Library Canberra	Doc Data Sheet

Capability Development Division

Director General Maritime Development	Doc Data Sheet
Director General Capability and Plans	Doc Data Sheet
Assistant Secretary Investment Analysis	Doc Data Sheet
Director Capability Plans and Programming	Doc Data Sheet
Director General Australian Defence Simulation Office	Doc Data Sheet

Chief Information Officer Group

Director General Australian Defence Simulation Office	Doc Data Sheet
Director General Information Policy and Plans	Doc Data Sheet
AS Information Strategy and Futures	Doc Data Sheet
AS Information Architecture and Management	Doc Data Sheet
Director General Information Services	Doc Data Sheet

Strategy Group

Director General Military Strategy	Doc Data Sheet
Assistant Secretary Governance and Counter-Proliferation	Doc Data Sheet

Navy

Maritime Operational Analysis Centre, Building 89/90 Garden Island Sydney NSW	Doc Data Sht & Dist List (shared)
Deputy Director (Operations)	
Deputy Director (Analysis)	
Director General Navy Capability, Performance and Plans, Navy Headquarters	Doc Data Sheet
Director General Navy Strategic Policy and Futures, Navy Headquarters	Doc Data Sheet

Air Force

SO (Science) - Headquarters Air Combat Group, RAAF Base, Williamtown NSW 2314	Doc Data Sht & Exec Summary

Army

ABCA National Standardisation Officer Land Warfare Development Sector, Puckapunyal	e-mailed Doc Data Sheet
SO (Science), Deployable Joint Force Headquarters (DJFHQ) (L), Enoggera QLD	Doc Data Sheet
SO (Science) - Land Headquarters (LHQ), Victoria Barracks NSW	Doc Data & Exec Summ

Information Capability Development

Director-General Information Capability Development	Doc Data Sheet
Director Command, Control Information Systems Development	1

Joint Operations Command

Director General Joint Operations	Doc Data Sheet
Chief of Staff Headquarters Joint Operations Command	Doc Data Sheet
SO Development ADF Warfare Centre	Doc Data Sheet
Director General Strategic Logistics	Doc Data Sheet
COS Australian Defence College	Doc Data Sheet
J68	1
J681	1

Intelligence and Security Group

DGSTA Defence Intelligence Organisation	1
Manager, Information Centre, Defence Intelligence Organisation	1 (PDF)
Assistant Secretary Capability Provisioning	Doc Data Sheet
Assistant Secretary Capability and Systems	Doc Data Sheet

Defence Materiel Organisation

Deputy CEO	Doc Data Sheet
Head Aerospace Systems Division	Doc Data Sheet
Head Maritime Systems Division	Doc Data Sheet
Head Electronic and Weapon Systems Division	Doc Data Sheet
Program Manager Air Warfare Destroyer	Doc Data Sheet

Defence Libraries

Library Manager, DLS-Canberra	1

Acquisitions Program

Director Project Development and Integration Command and Support Systems Branch	1
Project Director Joint Command Support System	1
Project Director Battlefield Command Support System	1
Project Director Joint Intelligence Support System	1

OTHER ORGANISATIONS

National Library of Australia	1
NASA (Canberra)	1
State Library of South Australia	1

Defence Industry

CAZ Computing	1
ADI	1
Tenix	1
SAAB	1
Tactica Data Systems	1

UNIVERSITIES AND COLLEGES

Australian Defence Force Academy

Library	1
Head of Aerospace and Mechanical Engineering	1
Hargrave Library, Monash University	Doc Data Sheet
Librarian, Flinders University	1

OUTSIDE AUSTRALIA

INTERNATIONAL DEFENCE INFORMATION CENTRES

US Defense Technical Information Center	1 PDF
UK Dstl Knowledge Services	1 PDF
Canada Defence Research Directorate R&D Knowledge & Information Management (DRDKIM)	1
NZ Defence Information Centre	1

ABSTRACTING AND INFORMATION ORGANISATIONS

Library, Chemical Abstracts Reference Service	1
Engineering Societies Library, US	1
Materials Information, Cambridge Scientific Abstracts, US	1
Documents Librarian, The Center for Research Libraries, US	1

SPARES	5

Total number of copies: Printed: 43 PDF: 3 46

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION
DOCUMENT CONTROL DATA

1. PRIVACY MARKING/CAVEAT (OF DOCUMENT)

2. TITLE

ORBAT Services Design Version 1.0

3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION)

Document	(U)
Title	(U)
Abstract	(U)

4. AUTHOR(S)

Glen Coomber, Carly Forbes and Philip Hawthorne

5. CORPORATE AUTHOR

DSTO *Defence Science and Technology Organisation*
PO Box 1500
Edinburgh South Australia 5111 Australia

6a. DSTO NUMBER	6b. AR NUMBER	6c. TYPE OF REPORT	7. DOCUMENT DATE
DSTO-TR-1727	AR-013-413	Technical Report	May 2005

8. FILE NUMBER	9. TASK NUMBER	10. TASK SPONSOR	11. NO. OF PAGES	12. NO. OF REFERENCES
E9505/25/224	JTW01/322	DGCSS	103	0

13. URL on the World Wide Web

http://www.dsto.defence.gov.au/corporate/reports/DSTO-TR-1727.pdf

14. RELEASE AUTHORITY

Chief, Command and Control Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for public release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS Yes

18. DEFTEST DESCRIPTORS

Order of Battle
Systems engineering
Combat support systems

19. ABSTRACT

The ORBAT Services Working Group was a joint DSTO, DMO and Industry activity to design, document and prototype a technical information infrastructure for applications and services within the Defence Information Environment that manipulate and store Order Of Battle (ORBAT) information. This document records the results of this activity and provides standardised Web Service Description Language interfaces, XML data representations and Use Cases for the management and manipulation of ORBAT information for use in Command, Control and Intelligence systems.